POLITECNICO DI TORINO

Corso di Laurea Magistrale in Ingegneria Meccanica

Tesi di Laurea Magistrale

Strategie di guida autonoma in retromarcia per robot mobili articolati

Relatore

Giuseppe Quaglia

Co-relatore

Andrea Botta

Candidato

Eleonora Moreno

Dicembre 2021

SOMMARIO

Analisi cinematica, dinamica e controllo dei robot mobili articolati Epi.q ed Agri.q, realizzati dal Dipartimento di Ingegneria Meccanica del Politecnico di Torino, al fine di sviluppare delle strategie di guida autonoma in retromarcia per poter evitare il fenomeno di instabilità. La strategia di controllo per giuda autonoma si compone delle fasi principali di pianificazione della traiettoria con o senza ostacoli, ottimizzazione della traiettoria per evitare fenomeni di instabilità, architettura di controllo per l'inseguimento della traiettoria garantendo la stabilità del sistema. Una serie di simulazioni cinematiche e dinamiche permettono la valutazione e la validazione della strategia sviluppata.

Indice

Indice delle Figure

Indice delle Tabelle

1. Capitolo 1: Introduzione

Negli ultimi decenni, il progresso scientifico e tecnologico ha favorito lo sviluppo di una grande varietà di robot mobili. L'utilizzo di veicoli autonomi sta diventando di rilevante interesse in diversi campi di applicazione; esistono soluzioni che vengono già impiegate per compiti di sicurezza, nella sorveglianza, per operare in ambienti pericolosi o per svolgere attività domestiche. Assistenza personale, agricoltura di precisione e soccorso sono solo alcuni esempi accanto alle più comuni applicazioni industriali. La pluralità dei possibili campi di applicazione ha portato alla necessità di rendere i robot impiegabili in spazi molto diversi tra loro. Spesso sono necessari progetti specifici per soddisfare le esigenze dettate da particolari spazi di lavoro: dislivelli del terreno, dimensione e presenza di ostacoli sono solo alcune delle caratteristiche che devono essere valutate per rendere quanto più possibile efficace e autonomo un robot mobile nel suo spazio di lavoro. Insieme alla crescita dei campi di applicazione della robotica mobile, ai robot sono richieste prestazioni nuove e migliori: le capacità di navigazione fuoristrada, le manovre rapide ad alta velocità e le particolari architetture di locomozione richiedono una profonda comprensione di come si comporta il robot dal punto di vista cinematico e dinamico per sviluppare un'adeguata pianificazione e controllo della traiettoria. Non a caso l'applicazione della navigazione autonoma ai robot mobili è una tendenza in aumento e, all'interno di questo campo, la pianificazione ed il controllo della traiettoria sono tra i temi più studiati [1-3].

Tra i design progettuali già proposti dalla comunità scientifica [4,5], la classe dei robot mobili articolati (o trattore-rimorchio) ha catturato particolare interesse per l'intrinseca modularità e versatilità che li contraddistingue e che porta spesso a preferirli rispetto ai robot monotelaio. Questi robot possono essere dotati di unità di locomozione particolarmente adatte a superare ostacoli e a muoversi su terreni sconnessi. Tali veicoli sono composti da due (o più di due) moduli, funzionalmente identici dal punto di vista della mobilità. Queste macchine sono in genere costruite come un modulo anteriore attivo e uno posteriore che viene tirato passivamente o che può contribuire alla trazione del veicolo quando richiesto [6]. La Figura 1.1 rappresenta schematicamente un robot mobile articolato a due moduli.

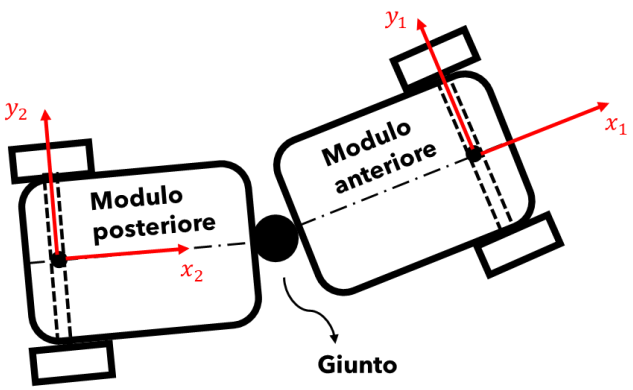

Figura 1.1 – Modello di un robot mobile articolato a due moduli

Benché la modellizzazione dei robot non sia nuova, molti lavori si sono concentrati sulla principale tipologia di robot mobili composti da un singolo modulo come i robot a guida differenziale [7] o i robot car-like [8]. Pochi studi approfondiscono i veicoli multi-modulari, anche se l'interesse nello sviluppo di autoarticolati autonomi ha contribuito alle ricerche sulla modellazione di robot mobili articolati. Tale interesse è dovuto al fatto che guidare e controllare i sistemi trattore-rimorchio è più complicato dei veicoli a corpo unico. Un problema che si presenta durante il movimento dei veicoli articolati è il cosiddetto fenomeno del jackknifing. Si tratta di una situazione in cui l'angolo di imbardata tra motrice e rimorchio cresce in modo tale che il veicolo si ripiega su sé stesso, diventando incontrollabile e causando possibili collisioni. Il jackknifing è un problema serio durante le manovre di retromarcia: mentre in marcia avanti il rimorchio viene trascinato dalla motrice, in marcia indietro viene spinto, il che rende instabile e difficilmente controllabile il comportamento del rimorchio. Quando si frena o si svolta bruscamente, è possibile che il fenomeno si manifesti anche in marcia avanti. La natura del fenomeno è però diversa nei due casi: nel moto in avanti è un effetto dinamico (anzi, inerziale) ed avviene solo ad alta velocità, mentre nel moto all'indietro il jackknifing è un problema cinematico che può manifestarsi anche a velocità molto basse. Ciò rende l'arretramento un tipico problema per i veicoli provvisti di rimorchio, poiché il rischio di perdita del controllo e di collisione con la motrice diventa probabile anche a basse velocità (Figura 1.2).

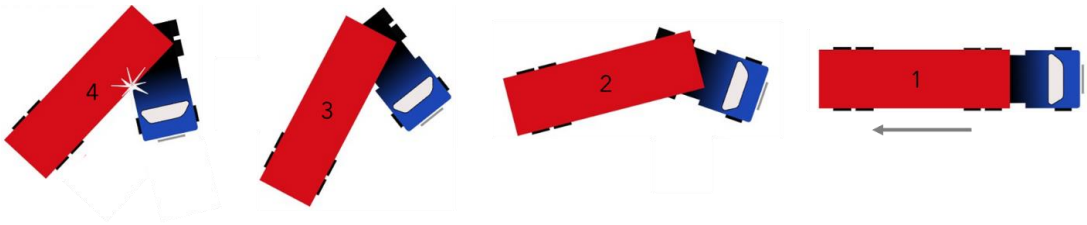

Figura 1.2 – Fenomeno del jackknifing durante la retromarcia di un veicolo con rimorchio

Molti prototipi di robot articolati interessanti sono già stati sviluppati o sono oggetto di studi attuali ed in letteratura sono presenti numerosi lavori che hanno proposto schemi di controllo per sistemi trattore-rimorchio [9-14] dove la maggior parte della ricerca è stata dedicata alle complesse problematiche di pianificazione del movimento cinematico derivanti dal movimento all'indietro e dall'evitamento degli ostacoli. Sebbene questi studi possano fornire alcune informazioni sul comportamento dei robot mobili articolati, la maggior parte della conoscenza proviene da studi sui veicoli convenzionali [15,16], cioè sui veicoli articolati car-like (Figura 1.3 a destra), mentre i robot che sono presi in considerazione in questo lavoro adottano una trazione differenziale (Figura 1.3 a sinistra).

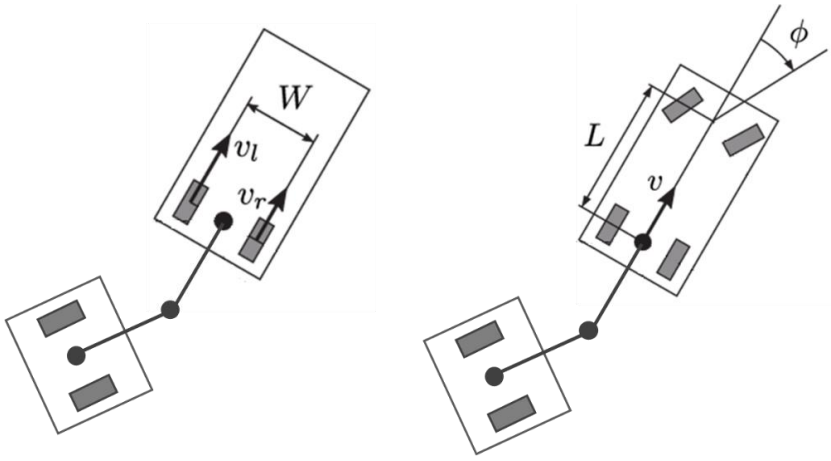

Figura 1.3 - Schema di un robot mobile articolato con trazione differenziale (a sinistra) e di un robot mobile articolato car-like (a destra)

Pertanto, questo studio si concentra sul jackknifing e le opportune strategie per evitarlo, prendendo in considerazione due veicoli articolati a guida differenziale che differiscono per dimensioni e scopo: i robot Epi.q ed Agri.q (Figura 1.4). Questi robot, sviluppati presso il Politecnico di Torino, sono due esempi di robot articolati, il primo progettato per compiti di sorveglianza, il secondo per compiti di agricoltura di precisione. Questi veicoli articolati sono composti da due parti principali, un modulo anteriore ed un modulo posteriore, tra loro

collegati tramite un giunto. Sul modulo anteriore sono poste ruote motrici non sterzanti, azionate da motori elettrici; due unità di locomozione analoghe sono montate su un asse rigido del carrello posteriore e possono essere attivate all'occorrenza. Una descrizione più approfondita dei robot si può trovare nel Capitolo 2.

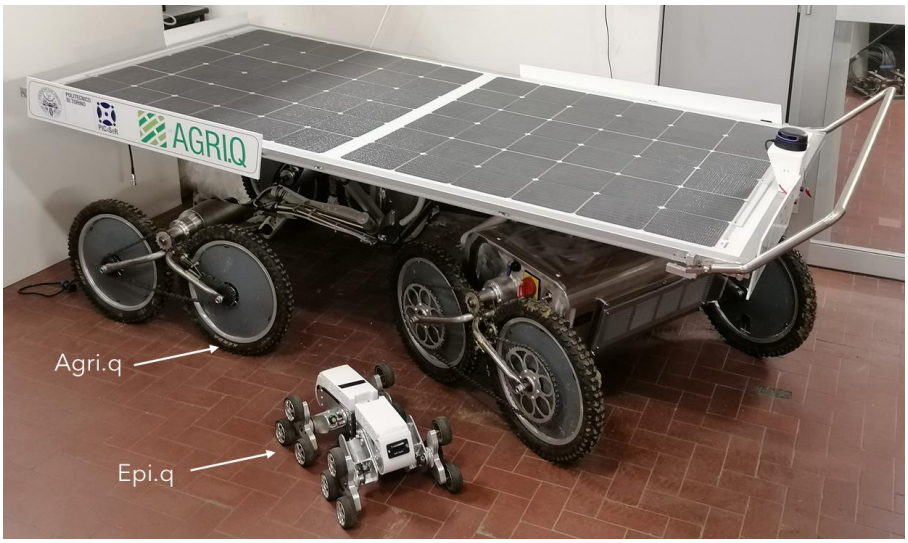

Figura 1.4 – I robot Epi.q ed Agri.q

Questo lavoro nasce perciò dall'esigenza di realizzare un sistema di controllo in retromarcia che guidi questi robot articolati lungo una traiettoria di riferimento evitando la divergenza dell'angolo di "attacco" (hitch angle) tra motrice e rimorchio. Lo studio viene organizzato nel modo seguente. L'osservazione di partenza è che il fenomeno del jackknife è una manifestazione dell'instabilità del hitch angle nel moto all'indietro (Capitolo 3) la quale causa la divergenza di quest'ultimo. Da questa osservazione nasce l'idea di progettare un controllore di basso livello con lo scopo di stabilizzare tale angolo. Questo approccio viene ripreso anche in [17]. Parallelamente, nei Capitoli 4 e 5 viene introdotto un sistema di controllo basato sul fatto che, durante la retromarcia, è conveniente spostare l'attenzione dal modulo anteriore a quello posteriore per una pianificazione efficace della traiettoria del robot. Una logica analoga viene anche utilizzata in [18] e [19]. Nei Capitoli 6 e 7 viene invece spostata l'attenzione sulla traiettoria andando ad individuare quali percorsi sono realizzabili da un robot articolato in marcia indietro, con o senza ostacoli. Poiché l'intervento del controllo sull'hitch angle porta il robot a discostarsi dalla traiettoria desiderata, nel Capitolo 8 viene proposto un sistema di inseguimento del percorso, basato sull'algoritmo di inseguimento geometrico noto come Pure Pursuit, appositamente riadattato per guidare un

veicolo trattore-rimorchio lungo una generica traiettoria all'indietro. Al termine di questo processo vengono riassunti i metodi di controllo per la guida autonoma in retromarcia sviluppati per i robot Epi.q ed Agri.q, accompagnati da alcune simulazioni agenti sui rispettivi modelli cinematici. Nel Capitolo 9 il controllo proposto per il rover Agri.q viene testato su una simulazione dinamica in ambiente Adams.

2. Capitolo 2: I robot mobili articolati Epi.q ed Agri.q

In questo capitolo viene riportata una breve descrizione dei robot mobili articolati Epi.q ed Agri.q. Si tratta di veicoli articolati costituiti da un modulo anteriore attivo ed uno posteriore che viene tirato passivamente o che può contribuire alla trazione del veicolo quando richiesto. In questo studio si considera l'avantreno come l'unico dotato di ruote motrici in quanto la locomozione dei robot articolati in retromarcia è solitamente a carico del modulo anteriore, mentre gli altri moduli vengono spinti.

2.1 Epi.q

Epi.q è un prototipo di robot mobile di piccole dimensioni, progettato e realizzato da un team del Politecnico di Torino – Dipartimento di Ingegneria Meccanica e Aerospaziale. Si tratta di un dispositivo robotico di trasporto in grado di muoversi in ambienti strutturati o non strutturati ed utilizzabile in diversi campi di applicazione: attività di monitoraggio, sorveglianza e intervento in spazi ristretti o in ambienti potenzialmente pericolosi. Ciò è reso possibile da una notevole mobilità, facilità di controllo e buona efficacia nell'attraversamento delle barriere. Inoltre, grazie all'approccio modulare che è stato adottato, Epi.q può essere rapidamente modificato in base alle specifiche del compito ottenendo così una struttura flessibile che può essere adattata a diversi compiti o requisiti. La Figura 2.1 mostra due foto di Epi.q.

Figura 2.1 – Epi.Q, UGV modulare di sorveglianza, visto frontalmente (a sinistra) e dall'alto (a destra)

2.1.1 Sistema di locomozione

Negli anni la letteratura scientifica ha presentato diverse soluzioni di locomozione per consentire ad un robot di muoversi in modo flessibile e affidabile su differenti terreni. I robot più comuni sono quelli a ruote, cingolati o dotati di gambe [20]. In generale, le soluzioni su ruote hanno una maggiore efficienza energetica, i robot cingolati sono molto controllabili, anche su terreni sconnessi, mentre quelli provvisti di gambe sono in grado di svolgere compiti più complessi. Esistono inoltre robot ibridi che cercano di comprendere i vantaggi di diverse classi di robot e, allo stesso tempo, di ridurre gli svantaggi. Epi.q appartiene a quest'ultima categoria: utilizzando un sistema di locomozione che unisce la locomozione su ruote e gambe, Epi.q può coniugare i vantaggi delle due modalità: alta velocità ed efficienza energetica su terreni pianeggianti con la locomozione su ruote, possibilità di movimento su terreni irregolari e in presenza di ostacoli con la locomozione a gambe, mostrando una grande capacità di muoversi in ambienti strutturati e non.

La commutazione tra locomozione su ruote e locomozione su gambe avviene in modo passivo, in base alle condizioni dinamiche e all'attrito locale, senza intervento di controllo, sfruttando i meccanismi epicicloidali dell'unità di locomozione. Questo aspetto semplifica il controllo e di conseguenza riduce il costo del robot. Inoltre, dal punto di vista energetico, l'approccio adottato appare favorevole rispetto ai sistemi di locomozione a gambe aumentandone l'autonomia operativa.

2.1.2 Architettura

Il robot Epi.q è composto da due moduli collegati tramite un giunto rotoidale passivo a due gradi di libertà (Figura 2.2). Il modulo anteriore (o avantreno) è costituito da uno scatolato che alloggia due motori, per comandare indipendentemente le unità di locomozione destra e sinistra, ed il pacco batteria. I motori (motoriduttori) sono situati nei braccetti laterali e, mediante una coppia conica con rapporto 1:1, trasmettono potenza alla ruota solare dell'unità motrice. Il modulo posteriore (chiamato anche retrotreno o rimorchio) contiene tutta l'elettronica di controllo e trasmissione, inoltre comprende due unità di locomozione inattive, ogni unità è costituita da tre ruote folli disposte radialmente.

L'elemento di collegamento tra la parte anteriore e posteriore del robot è un giunto a 2 gradi di libertà perpendicolari tra loro che permette imbardata e rollio relativi; il grado di libertà di rollio garantisce un corretto contatto tra le ruote ed il terreno anche in presenza di terreni sconnessi, anche senza un sistema di sospensione. L'escursione angolare di questo giunto può essere limitata mediante arresti meccanici: se si considera ad esempio il prototipo Epi.q-TG il giunto verticale consente un angolo di sterzata limitato a $\pm 55°$.

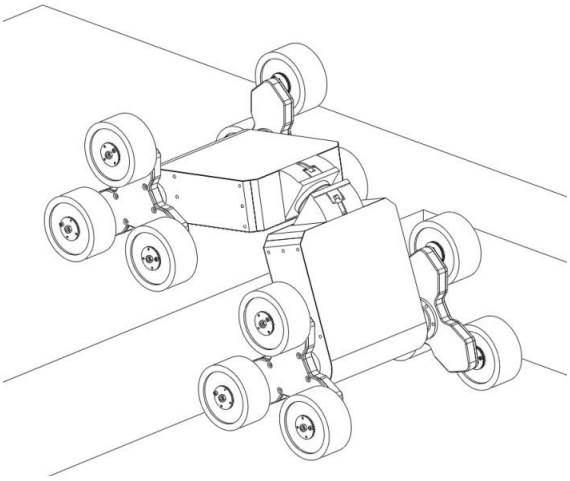

Figura 2.2 - Rappresentazione di Epi.q Lizard
mentre supera un gradino

2.1.3 Guida differenziale

I robot Epi.q sfruttano uno sterzo differenziale, che fornisce sia le funzioni di guida che di sterzo. Un veicolo a sterzo differenziale è costituito da due o più ruote montate lungo lo stesso asse, azionate e comandate indipendentemente.

Se entrambi i motori, uniti alle unità motrici, vengono azionati nella stessa direzione e velocità, il robot procede in linea retta. Se un'unità motrice ruota più velocemente dell'altra, il robot segue un percorso curvo, girando verso l'unità motrice più lenta. Se una delle unità motrici viene fermata mentre l'altra continua a girare, l'avantreno ruota attorno all'unità motrice arrestata. Se le unità motrici girano a velocità uguale ma in direzioni opposte, entrambe le unità motrici percorrono una traiettoria circolare attorno ad un punto centrato a metà tra le due unità motrici, quindi l'avantreno ruota attorno all'asse verticale. Questo vuol

dire che, scegliendo le velocità delle unità motrici, si agisce in modo tale che il robot possa seguire la traiettoria desiderata.

2.1.4 Driving unit

L'unità motrice è la caratteristica principale di Epi.q. Come visibile in Figura 2.2, il robot è dotato di quattro unità motrici, ciascuna portante tre ruote radiali montate all'estremità di ciascun raggio. Il sistema di trasmissione è basato su un sistema di ingranaggi epicicloidale azionato da un singolo motore elettrico accoppiato all'ingranaggio solare, mentre le ruote del robot sono collegate con tre ingranaggi (i satelliti del riduttore epicicloidale).

L'unità motrice è progettata per avere una coppia limite che attiva le due modalità di locomozione: se la coppia richiesta per muoversi su ruote supera la coppia richiesta per muoversi su gambe, il robot cambia di conseguenza la sua locomozione, da rotolamento su ruote al moto su gambe, e viceversa. Pertanto, è necessario un solo motore per unità motrice per entrambi i tipi di locomozione.

Quando il robot è in movimento su ruote (modalità di avanzamento) il peso del robot e il contatto tra le ruote e il suolo vincolano la posizione angolare dell'unità motrice (Figura 2.3 a sinistra). In particolare, durante il movimento su superfici piane, il peso del robot impedisce la rotazione del portatreno. Quando il robot urta un ostacolo, se l'attrito locale tra la ruota anteriore e l'ostacolo è sufficiente per arrestare la rotazione della ruota, la rivoluzione del portatreno inizia automaticamente, l'unità motrice inizia a ruotare attorno al centro della ruota ferma. Questo porta un'altra ruota a contatto con la superficie dell'ostacolo, consentendo al robot di scavalcare piccoli ostacoli senza comandi o azioni esterne (modalità di salita automatica), utilizzando il portatreno come un'unità rotante a gambe, come mostrato nella Figura 2.3 a destra. Quindi, ogni unità di locomozione ha due gradi di libertà ed è in grado di auto-riconfigurare la sua cinematica in base allo schema di vincolo.

Ulteriori dettagli sul funzionamento della famiglia dei robot Epi.q possono essere trovati in [6, 21-24].

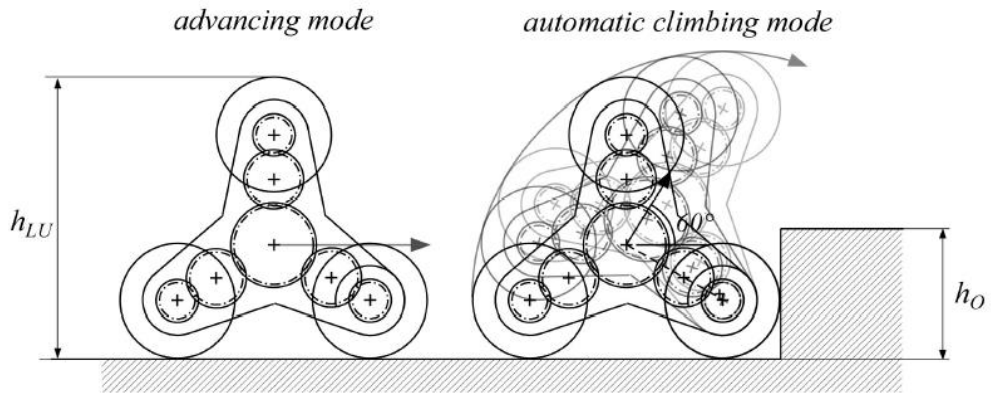

Figura 2.3 – Advancing mode e automatic climbing mode

2.2 Agri.q

L'agricoltura di precisione [25, 26] è una strategia di gestione volta a migliorare la produttività, la redditività e la sostenibilità della produzione agricola grazie all'incorporazione di progressi tecnologici sviluppati principalmente per altri settori. Tra questi, la robotica sta svolgendo un ruolo di primo piano poiché consente di monitorare puntualmente i fattori che influenzano i processi produttivi, inoltre i robot possono essere dotati di strumenti o end effector per eseguire diverse azioni in modo da soddisfare specifiche esigenze [27-29]. In questo scenario i ricercatori del Politecnico di Torino hanno sviluppato un innovativo veicolo terrestre senza pilota (UGV - Unmanned Ground Vehicle) concepito per lo svolgimento di compiti di monitoraggio delle colture e per il campionamento di suolo e foglie, con particolare attenzione al settore vitivinicolo. Il rover, denominato Agri.q, è un robot mobile a otto ruote. È un veicolo elettrico progettato per operare in ambienti non strutturati su terreni irregolari, per muoversi tra i filari e per cooperare con droni. Il robot, come mostrato in Figura 2.4, è dotato di un braccio robotico, per svolgere le attività di campionamento, e di pannelli solari volti a migliorare in modo sostenibile la durata della batteria. Nello specifico, Agri.q presenta una piattaforma di atterraggio a due gradi di libertà in grado di auto-orientarsi per garantire un attracco sicuro del drone o per massimizzare la raccolta dei raggi solari durante la fase di auto-ricarica. In questo modo vengono aumentate l'autonomia e la sostenibilità del rover.

Figura 2.4 – Prototipo del robot mobile Agri.q

2.2.1 Design funzionale e meccanico

Per il sistema di locomozione è stato impiegato un approccio modulare. L'idea portante che ha guidato la progettazione del robot Agri.q è l'efficienza della locomozione su terreni irregolari. Una delle soluzioni più efficaci a tale problema è l'adozione di cingoli in quanto possono distribuire il peso del veicolo su un'ampia superficie permettendo al rover di affrontare le asperità del terreno anche in condizioni di superfici bagnate o scivolose. Lo svantaggio principale rispetto ai veicoli a ruote è un'efficienza estremamente ridotta. Il sistema di trazione di Agri.q è concepito come un buon compromesso tra l'efficienza delle ruote e l'efficacia dei cingoli su superfici irregolari, sia dal punto di vista energetico che della mobilità. A tal fine, il rover è stato dotato di quattro unità motrici, due anteriori e due posteriori, ciascuna composta da due ruote alloggiate su un bilanciere. Uno schema del robot è mostrato in Figura 2.5a. Il collegamento di ciascun bilanciere con il corpo principale del robot avviene tramite un giunto rotoidale passivo, la cui libera rotazione consente alle ruote di seguire le pendenze del terreno indipendentemente dalle configurazioni degli altri bilancieri (Figura 2.5b).

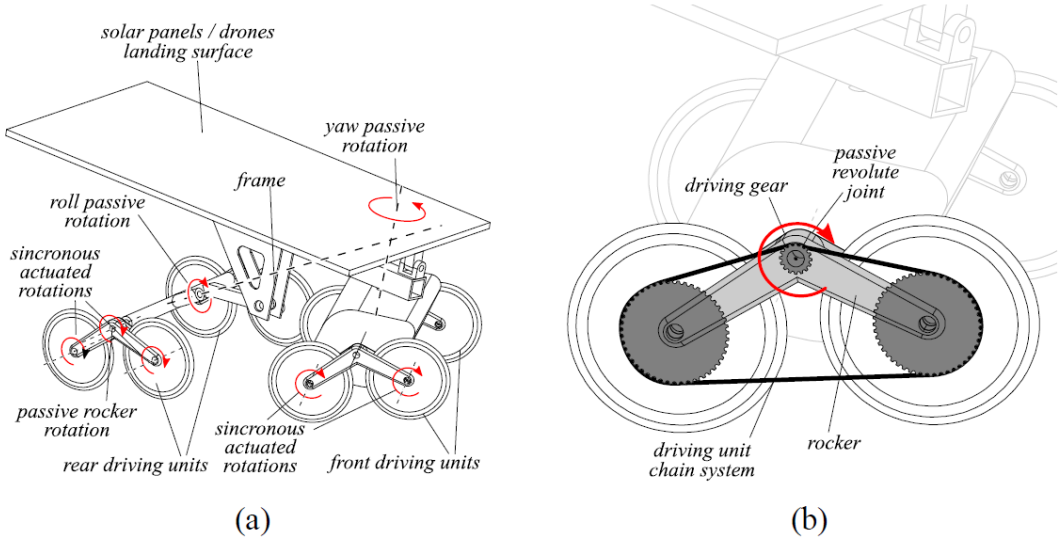

Figura 2.5 – Schema funzionale del rover Agri.q (a), connessione del bilanciere con il modulo anteriore del robot (b)

L'elevato numero di punti di contatto con il suolo garantisce una distribuzione del peso simile a quella di un robot cingolato, pur mantenendo un'efficienza più vicina a quella di una macchina su ruote; il bilanciere può ruotare attorno al giunto rotoidale per garantire una corretta distribuzione delle forze normali agenti al suolo su un'ampia superficie di contatto (Figura 2.5b). Di conseguenza, si evita che il veicolo rimanga bloccato in un terreno cedevole e, allo stesso tempo, si evita la compattazione del terreno. Tuttavia, l'intera efficienza di trazione rimane simile a quella di un rover a quattro ruote. Un ulteriore vantaggio deriva dalla capacità del bilanciere di agire da filtro rispetto alle oscillazioni imposte dal terreno: lo spostamento verticale imposto alla ruota da una notevole irregolarità del terreno si converte in uno spostamento minore del giunto del bilanciere e quindi dell'intero veicolo (Figura 2.7).

Un ulteriore miglioramento dell'efficienza è stato ottenuto dotando il robot di una rotazione passiva di imbardata tra i bilancieri anteriori e posteriori, permettendo una significativa riduzione dello slittamento laterale delle ruote rispetto al suolo durante una traiettoria curva rispetto ad un robot a sterzatura differenziale composto da un unico modulo e con numero simile di ruote. Inoltre, è stato aggiunto un grado di libertà di rollio passivo per migliorare la capacità di adattamento del rover alle irregolarità del suolo. Questo giunto rotoidale permette infatti le rotazioni reciproche dei bilancieri anteriore e posteriore, garantendo la loro capacità di mantenere un contatto stabile delle ruote con il terreno (Figura 2.6).

Figura 2.6 – Agri.q durante la simulazione di un terreno accidentato

Figura 2.7 – Unità motrice di Agri.q

Come accennato in precedenza, il rover è stato dotato di due celle solari, in grado di ricaricare le batterie del rover. La dimensione del pannello solare è stata scelta per garantire la ricarica della batteria durante i tempi di inattività tra le missioni e di limitare la scarica delle stesse durante l'attività. Tali pannelli hanno il duplice scopo di fornire un'efficiente fonte di energia per il robot e di essere utilizzati come piattaforma di atterraggio dagli UAV (Unmanned Aerial Vehicles). In entrambi i casi il piano superiore del rover viene orientato secondo necessità sfruttando due gradi di libertà (beccheggio e rollio) attorno a due assi indipendenti. Ciò consente di massimizzare la raccolta dei raggi solari durante la fase di ricarica, oltre a fornire una superficie di atterraggio sempre orizzontale per i droni.

Gli attuatori dei sistemi di locomozione modulari sono stati scelti considerando, come condizione più gravosa, una pendenza di circa 15°, da superare con i soli due motori di trazione anteriori, ad una velocità massima di circa 5 km /h. Per raggiungere tale obiettivo, la trasmissione di ogni unità è composta da un motore elettrico DC brushed, collegato ad un riduttore epicicloidale commerciale. Un'ulteriore riduzione di velocità è poi fornita da un sistema a catena tra il motoriduttore e la coppia di ruote dello stesso lato di entrambi i moduli. Inoltre, il rover è dotato di due unità di locomozione posteriori che possono essere attivate in caso di salite particolarmente ripide, curve molto strette o in generale quando serve più coppia e quindi è richiesto il passaggio dell'architettura da 2 unità a 4 unità motrici; in questo secondo caso la trasmissione della potenza è distribuita alle otto ruote a contatto con il terreno.

Per maggiori dettagli sulla struttura del robot si rimanda a [30-34].

2.3 Differenze tra Epi.q ed Agri.q

Nonostante la differenza di scala rilevante, i due robot differiscono per pochissimi dettagli funzionali [35]:

- **posizione del giunto centrale**: i due robot sono caratterizzati da una diversa posizione del giunto di collegamento tra il modulo anteriore e posteriore;
- **numero di ruote**: Epi.q è un piccolo robot mobile articolato, pensato appositamente per superare ostacoli di diversa dimensione e natura; Agri.q è un rover agricolo, il cui sistema di trazione è stato ottimizzato per affrontare le irregolarità del suolo. Per questo Epi.q è stato dotato di un totale di dodici ruote, tre per ogni unità motrice, per migliorarne la capacità di salita. Tuttavia, durante il normale utilizzo entrambi i robot possono contare su otto ruote contemporaneamente a contatto col suolo.

Tenendo presenti le peculiarità e le differenze di Epi.q e Agri.q, le sezioni successive illustrano un modello cinematico generale in grado di esprimere il comportamento dei due robot dopo l'identificazione dei rispettivi parametri.

3. Capitolo 3: Modello cinematico

Sebbene i robot Epi.q ed Agri.q siano costruiti con scale estremamente diverse (il primo è un robot di piccola taglia di circa 5 kg e 30 cm di lunghezza, mentre il secondo pesa 100 kg ed è lungo quasi 2 m), queste due macchine condividono un'architettura cinematica comune. Tenendo presenti le peculiarità e le differenze di Epi.q ed Agri.q, è possibile sviluppare un modello cinematico generale in grado di esprimere il comportamento dei due robot.

Questi robot mobili possono essere analizzati sotto diversi punti di vista a seconda dei molteplici aspetti che si vogliono mettere in risalto. In quest'ottica vengono introdotti due modelli cinematici equivalenti, che permettono di mettere in evidenza diverse caratteristiche di un robot mobile articolato.

Il primo modello cinematico [36] consiste nell'esprimere lo spazio delle configurazioni del robot mobile, vale a dire l'insieme degli stati necessari per esprimere la configurazione del robot sul piano euclideo (x_0, y_0).

Il secondo modello cinematico invece, che verrà trattato nel capitolo successivo, va a considerare il robot articolato come un insieme di due corpi rigidi uniti tra loro mediante una cerniera J. Quindi per il modulo anteriore e posteriore può essere svolta un'analisi cinematica distinta.

In questo capitolo viene trattata l'analisi del primo modello cinematico indicato. Tale modello è basato sulle seguenti ipotesi:

- Il modello è puramente cinematico.
- Il modello considera solo il movimento su una superficie piana, movimenti al di fuori del piano non vengono considerati.
- I moduli di locomozione sono semplificati considerando una ruota singola equivalente per ogni lato con asse passante per il punto di riferimento del modulo (O_1 o O_2).
- Ogni ruota rotola senza strisciare, cioè senza slittare né longitudinalmente né trasversalmente.

Prima di procedere con lo studio cinematico, si introduce un sistema di riferimento piano fisso (x_0, y_0) e gli angoli φ_1 e φ_2 che rispettivamente modulo anteriore e modulo posteriore formano rispetto all'asse x_0 (positivi in senso antiorario)[1].

In Figura 3.1 viene mostrata la struttura cinematica del robot a otto ruote in una rappresentazione planare semplificata. Ognuno dei due moduli è caratterizzato da una coppia di unità motrici ed ogni unità motrice ha due ruote a contatto con il suolo. La posizione dei due moduli, anteriore e posteriore, è descritta mediante due telai. Le origini dei telai (punto O_1 per l'anteriore, punto O_2 per il posteriore) si trovano entrambe in corrispondenza dell'intersezione tra l'asse di rotazione dei portanti delle unità motrici con l'asse longitudinale.

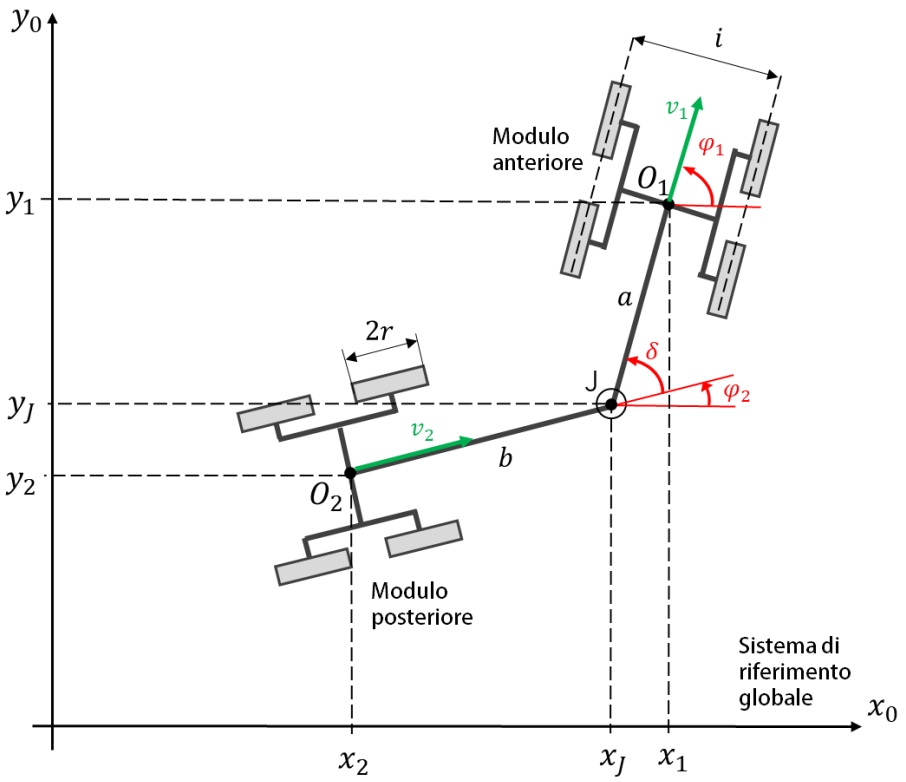

Figura 3.1 – Schematizzazione del robot mobile

La posa nel piano dei due moduli è determinata da un insieme di tre variabili ciascuno, due per la posizione ed uno per l'orientamento, come descritto nella Tabella 3.1.

[1] L'asse z_0, uscente dal piano (x_0, y_0), va a definire il verso positivo di rotazione sul piano (x_0, y_0).

Tabella 3.1 – Grandezze fondamentali di Agri.q ed Epi.q.

Simbolo	Descrizione
(x_1, y_1, φ_1)	Posa del modulo anteriore (avantreno)
(x_2, y_2, φ_2)	Posa del modulo posteriore (retrotreno o rimorchio)
δ	Rotazione relativa (imbardata) tra modulo anteriore e posteriore
a	Distanza tra il giunto centrale ed il modulo anteriore.
b	Distanza tra il giunto centrale ed il modulo posteriore
i	Interasse trasversale tra le ruote
r	Raggio delle ruote
v_1	Velocità longitudinale del modulo anteriore
$\dot{\varphi}_1$	Velocità angolare di imbardata del modulo anteriore
v_2	Velocità longitudinale del modulo posteriore

La Tabella 3.2 invece introduce i parametri geometrici che vengono utilizzati per il processo di modellazione dei due prototipi Epi.q ed Agri.q.

Tabella 3.2 – Parametri geometrici e cinematici di Epi.q ed Agri.q.

Simbolo	Epi.q	Agri.q	
a	0.132	0.000	[m]
b	0.139	1.300	[m]
i	0.260	0.845	[m]
r	0.032	0.195	[m]
δ_{max}, δ_{min}	$\pm 55°$	$\pm 35°$	-
$v_{1,max}$, $v_{1,min}$	± 1	± 1.5	[m/s]

Le relazioni geometriche tra i moduli impongono i seguenti vincoli

$$x_2 = x_1 - b\cos\varphi_2 - a\cos\varphi_1 \tag{3.1}$$

$$y_2 = y_1 - b\sin\varphi_2 - a\sin\varphi_1 \tag{3.2}$$

$$\delta = \varphi_1 - \varphi_2 \tag{3.3}$$

Ogni ruota introduce un vincolo anolonomo nel sistema in quanto non consente una traslazione normale alla direzione di rotolamento (non è progettata per scorrere lateralmente), per questa ragione la velocità normale al verso di rotolamento è nulla. Possono quindi essere definiti due ulteriori vincoli anolonomi che vanno a limitare la mobilità del veicolo in movimento

$$\dot{x}_1\sin\varphi_1 - \dot{y}_1\cos\varphi_1 = 0 \tag{3.4}$$

$$\dot{x}_2\sin\varphi_2 - \dot{y}_2\cos\varphi_2 = 0 \tag{3.5}$$

La velocità nel sistema di riferimento globale di ogni modulo è definita come

$$\dot{x}_1 = v_1 \cos \varphi_1$$

$$\dot{y}_1 = v_1 \sin \varphi_1$$

$$\dot{x}_2 = v_2 \cos \varphi_2$$

$$\dot{y}_2 = v_2 \sin \varphi_2$$

Lo spazio delle configurazioni del robot mobile, vale a dire l'insieme degli stati che servono per localizzare la configurazione del robot nel piano, viene espresso come

$$q = \begin{bmatrix} x_1 \\ y_1 \\ \varphi_1 \\ \delta \end{bmatrix} \tag{3.6}$$

Perciò il modello cinematico può essere scritto come

$$\dot{q} = \begin{bmatrix} \dot{x}_1 \\ \dot{y}_1 \\ \dot{\varphi}_1 \\ \dot{\delta} \end{bmatrix} = A \begin{bmatrix} v_1 \\ \dot{\varphi}_1 \end{bmatrix} \tag{3.7}$$

La relazione che lega $\dot{\delta}$ alle velocità v_1 e $\dot{\varphi}_1$ può essere ottenuta derivando (3.1), (3.2) e (3.3) nel tempo e sostituendo i risultati in (3.5). È così possibile ottenere

$$\dot{\delta} = \left(\frac{a}{b}\cos \delta + 1\right) \dot{\varphi}_1 - \frac{1}{b} \sin \delta \cdot v_1 \tag{3.8}$$

Quindi il modello cinematico può essere esplicitamente scritto come

$$\dot{q} = \begin{bmatrix} \dot{x}_1 \\ \dot{y}_1 \\ \dot{\varphi}_1 \\ \dot{\delta} \end{bmatrix} = \begin{bmatrix} \cos \varphi_1 & 0 \\ \sin \varphi_1 & 0 \\ 0 & 1 \\ -\frac{1}{b}\sin \delta & \frac{a}{b}\cos \delta + 1 \end{bmatrix} \begin{bmatrix} v_1 \\ \dot{\varphi}_1 \end{bmatrix} \tag{3.9}$$

Ipotizzando che i moduli possano essere semplificati considerando una singola ruota equivalente per ogni lato, è possibile affermare che ogni modulo del robot è a sterzatura differenziale con due sole ruote (Figura 3.2); quindi, le velocità longitudinali ed angolari di ogni modulo possono essere ricavate facilmente conoscendo le velocità angolari delle ruote.
Si procede col determinare la relazione che lega le velocità v_{1L} e v_{1R}, rispettivamente della ruota sinistra e destra del modulo frontale, alla velocità longitudinale v_1 (Figura 3.3).

$$v_1 = \frac{1}{2}(v_{1R} + v_{1L}) \tag{3.10}$$

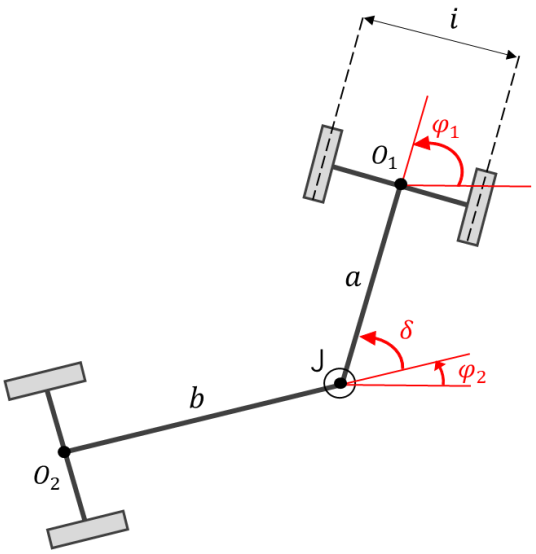

Figura 3.2 – Schematizzazione semplificata del robot mobile

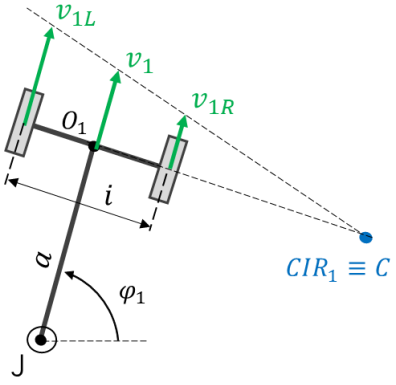

Figura 3.3 – Schematizzazione del modulo frontale e delle corrispondenti velocità

Inoltre, nota la distanza i delle ruote, ricordando che le velocità delle ruote sono assunte positive se in avanti e che il verso positivo di φ_1 è antiorario, si ricava la velocità angolare $\dot{\varphi}_1$:

$$v_{1L} = -\left(\overline{CO_1} + \frac{i}{2}\right)\dot{\varphi}_1$$

$$v_{1R} = -\left(\overline{CO_1} - \frac{i}{2}\right)\dot{\varphi}_1$$

$$v_{1L} = v_{1R} - i \cdot \dot{\varphi}_1$$

Quindi si ottiene:

$$\dot{\varphi}_1 = \frac{1}{i}(v_{1R} - v_{1L}) \tag{3.11}$$

L'ipotesi di ridurre le ruote ad una per ogni lato è fondamentale in quanto, se non venisse introdotta, si dovrebbe anche considerare la componente laterale di velocità delle ruote.

Note le relazioni (3.10) e (3.11), è possibile riscrivere l'espressione del modello cinematico in una forma alternativa:

$$\dot{q} = \begin{bmatrix} \dot{x}_1 \\ \dot{y}_1 \\ \dot{\varphi}_1 \\ \dot{\delta} \end{bmatrix} = \begin{bmatrix} \frac{1}{2}\cos\varphi_1 & \frac{1}{2}\cos\varphi_1 \\ \frac{1}{2}\sin\varphi_1 & \frac{1}{2}\sin\varphi_1 \\ \frac{1}{i} & -\frac{1}{i} \\ \frac{a\cos\delta + b - \frac{i}{2}\sin\delta}{bi} & -\frac{a\cos\delta + b + \frac{i}{2}\sin\delta}{bi} \end{bmatrix} \begin{bmatrix} v_{1R} \\ v_{1L} \end{bmatrix} \tag{3.12}$$

v_{1R} e v_{1L} possono essere facilmente legate alle velocità angolari delle ruote ω_{1R} e ω_{1L} conoscendo il raggio r della ruota.

$$v_{1R} = r \cdot \omega_{1R}$$

$$v_{1L} = r \cdot \omega_{1L}$$

Quindi le relazioni (3.10) e (3.11) possono essere riscritte come:

$$v_1 = \frac{r}{2}(\omega_{1R} + \omega_{1L}) \tag{3.13}$$

$$\dot{\varphi}_1 = \frac{r}{i}(\omega_{1R} - \omega_{1L}) \tag{3.14}$$

3.1 Raggio di curvatura del modulo anteriore

Per poter descrivere una traiettoria, rettilinea, circolare o curvilinea, che il robot dovrà seguire è utile introdurre il concetto di raggio di curvatura. Quando si studiano due corpi rigidi tra loro articolati bisogna considerare che, nel caso più generale, durante il moto essi percorrono due traiettorie distinte. Questo equivale a dire che ogni modulo è caratterizzato da un suo raggio di curvatura R e da un suo centro di istantanea rotazione CIR.

Poiché si considera che il modulo anteriore sia l'unico ad essere dotato di ruote motrici, si definisce la relazione che lega il raggio di curvatura R_1 del percorso seguito da tale modulo alla cinematica del robot.

Si inizia col determinare il centro di istantanea rotazione CIR_1: questo si trova all'incrocio del prolungamento dell'assale con la retta congiungente i vettori velocità delle due ruote.

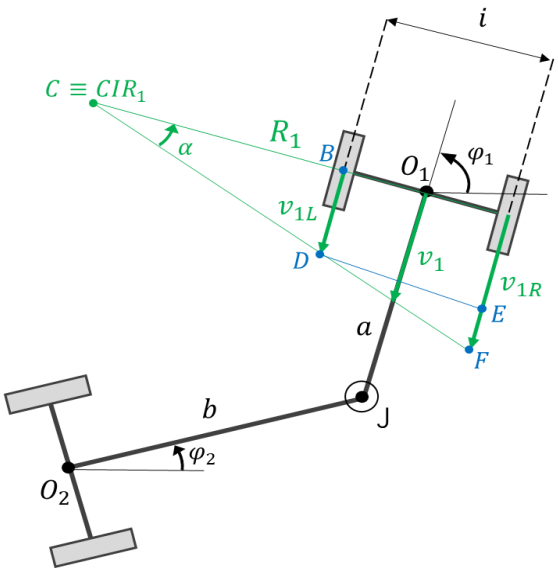

Figura 3.4 – Rappresentazione del raggio di curvatura dell'avantreno in sterzata

Osservando la Figura 3.4 si nota che

$$R_1 = \overline{CB} + \frac{i}{2}$$

Partendo da questa relazione geometrica si ricava l'equazione lega le velocità v_{1R} e v_{1L} delle ruote al raggio di curvatura R_1. Considerando il triangolo CBD

$$\overline{CB} = \frac{v_{1L}}{\tan\alpha}$$

Considerando invece il triangolo DEF

$$v_{1R} - v_{1L} = i\tan\alpha \qquad \tan\alpha = \frac{v_{1R} - v_{1L}}{i}$$

Quindi

$$R_1 = \frac{v_{1L}}{\tan\alpha} + \frac{i}{2} = \frac{v_{1L}}{\frac{v_{1R} - v_{1L}}{i}} + \frac{i}{2}$$

Si ricava così l'espressione del raggio di curvatura

$$R_1 = i\left(\frac{1}{2} + \frac{v_{1L}}{v_{1R} - v_{1L}}\right)$$

Essendo il raggio r uguale per tutte le ruote, si possono sostituire alle velocità tangenziali quelle angolari ottenendo:

$$R_1 = i\left(\frac{1}{2} + \frac{\omega_{1L}}{\omega_{1R} - \omega_{1L}}\right) = \frac{i}{2} \cdot \frac{\omega_{1R} + \omega_{1L}}{\omega_{1R} - \omega_{1L}} \tag{3.15}$$

Questa relazione mostra come il raggio di curvatura, e di conseguenza la traiettoria generata, dipenda dal rapporto delle velocità angolari delle ruote, mentre il valore delle singole velocità influisce sui tempi di percorrenza della traiettoria.

Noto il raggio di curvatura, la curvatura riferita all'avantreno è esprimibile come:

$$\rho_1 = \frac{1}{R_1} = \frac{2}{i} \cdot \frac{\omega_{1R} - \omega_{1L}}{\omega_{1R} + \omega_{1L}} \tag{3.16}$$

È importante sottolineare come le ultime due espressioni valgano sia in marcia avanti che in marcia indietro.

Partendo dalla relazione (3.15) è possibile fare alcune considerazioni. La prima osservazione riguarda le velocità del modulo anteriore: la velocità di un punto di un corpo rigido in rotazione è pari al prodotto della velocità angolare del corpo per la distanza tra il centro di istantanea rotazione del corpo ed il punto. Se si considera il punto O_1, la distanza tra CIR_1 ed O_1 corrisponde al raggio di curvatura R_1, quindi si può affermare che:

$$v_1 = R_1 \dot{\varphi}_1 = \frac{\dot{\varphi}_1}{\rho_1} \tag{3.17}$$

Questa relazione è estremamente importante perché permette di legare la velocità longitudinale alla velocità angolare del modulo anteriore mediante il raggio di curvatura di quest'ultimo.

La seconda considerazione riguarda il segno della curvatura ρ_1, infatti, la convezione adottata per ricavare questa formula prevede:

- Curvatura positiva ($|\omega_{1R}| > |\omega_{1L}|$) se il robot ruota in senso antiorario in marcia avanti (orario in marcia indietro).
- Curvatura negativa ($|\omega_{1R}| < |\omega_{1L}|$) se il robot ruota in senso orario in marcia avanti (antiorario in marcia indietro).
- Curvatura nulla se il robot percorre una traiettoria rettilinea ($|\omega_{1R}| = |\omega_{1L}|$).

3.2 Angolo di equilibrio del sistema

Analizzando l'equazione (3.8) si può notare che il robot presenta un certo angolo δ di equilibrio, cioè un valore di δ tale per cui $\dot{\delta} = 0$.

Il primo passo per evitare il fenomeno del jackknife consiste nel determinare questo angolo δ di equilibrio (il quale varia a seconda della traiettoria seguita dal robot). Esistono due casi possibili:

- Traiettoria rettilinea ($\dot{\varphi}_1 = 0$).
- Traiettoria circolare ($\dot{\varphi}_1 \neq 0$).

3.2.1 Traiettoria rettilinea

Per determinare l'angolo δ di equilibrio si pone l'equazione (3.8) uguale a 0.

$$\dot{\delta} = \left(\frac{a}{b}\cos\delta_{eq} + 1\right)\dot{\varphi}_1 - \frac{1}{b}\sin\delta_{eq} \cdot v_1 = 0$$

Se il robot va dritto ($\dot{\varphi}_1 = 0$) allora l'equazione si semplifica in:

$$-\frac{1}{b}\sin\delta_{eq} \cdot v_1 = 0$$

$$\sin\delta_{eq} = 0$$

Si ottengono così due soluzioni possibili:

$$\delta_{eq,1} = 0 \quad \text{oppure} \quad \delta_{eq,2} = \pi$$

La soluzione $\delta_{eq,1} = 0$ corrisponde all'avere modulo anteriore e posteriore allineati, mentre la soluzione $\delta_{eq,2} = \pi$ corrisponde all'avere modulo anteriore e posteriore sovrapposti.

Noto il modello cinematico dei robot, è possibile tradurre tale modello in un codice Matlab per eseguire delle simulazioni. Le seguenti simulazioni sono state effettuate per un veicolo articolato con le stesse caratteristiche cinematiche del robot Epi.q.

Robot: Epi.q

Tipo di moto:
marcia avanti,
traiettoria rettilinea

Condizioni iniziali
$\varphi_{1,0} = 0°$
$\delta_O = 35°$

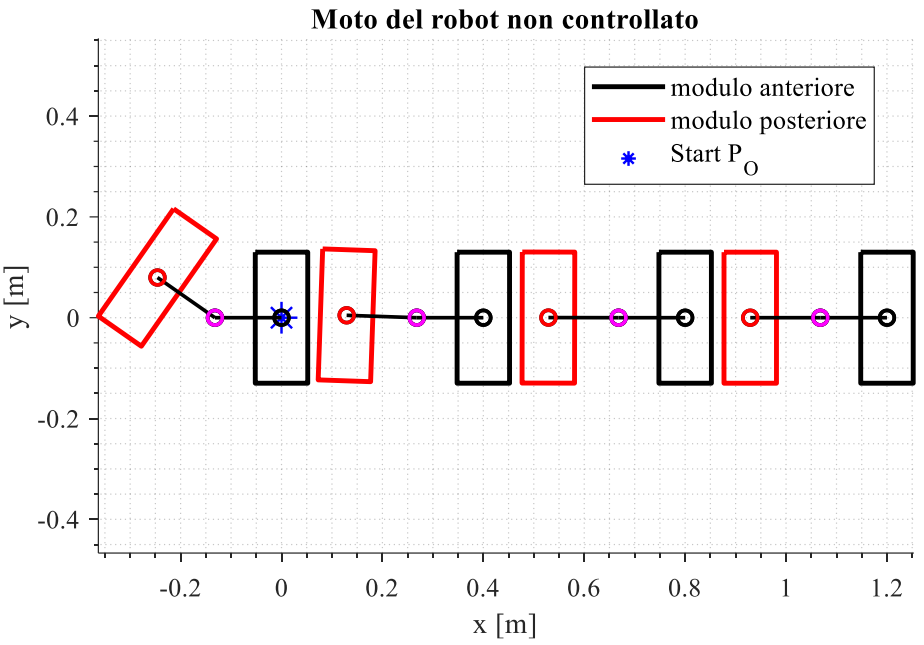

Figura 3.5 – Simulazione del moto in marcia avanti

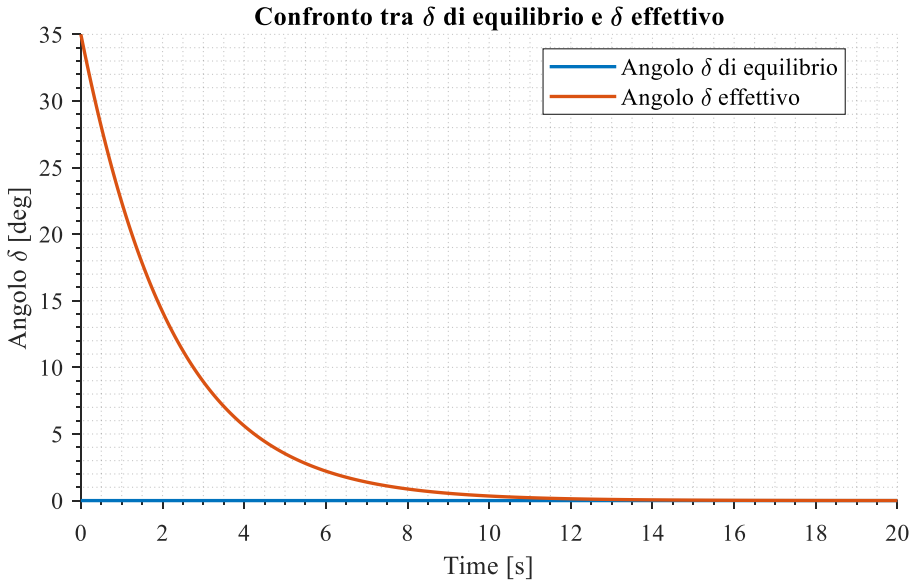

Figura 3.6 – Andamento dell'angolo δ lungo il tratto rettilineo

Robot: Epi.q

Tipo di moto: marcia indietro, traiettoria rettilinea

Condizioni iniziali:
$\varphi_{1,0} = 0°$
$\delta_O = 2°$

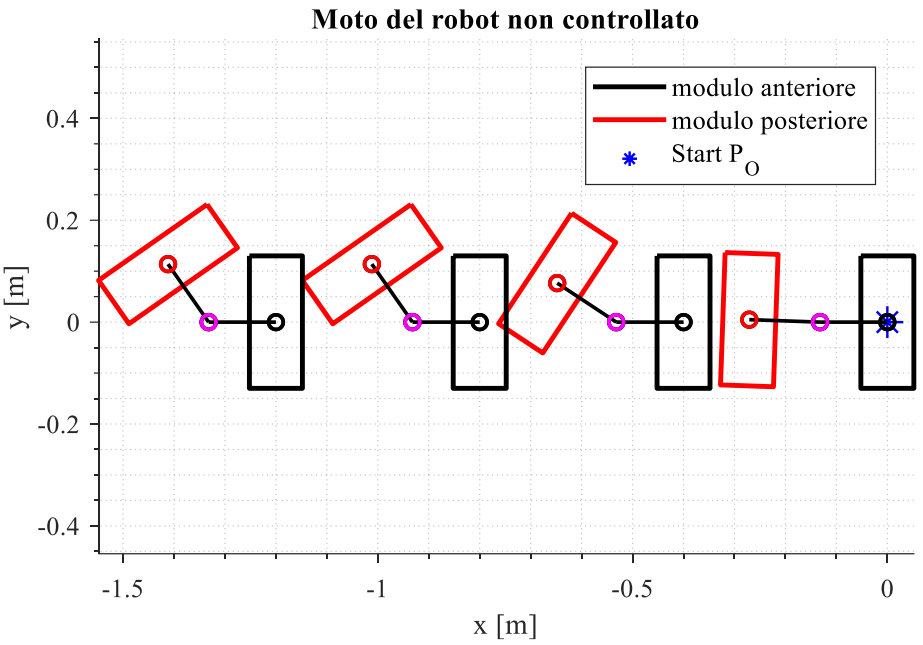

Figura 3.7 – Simulazione del moto in marcia indietro

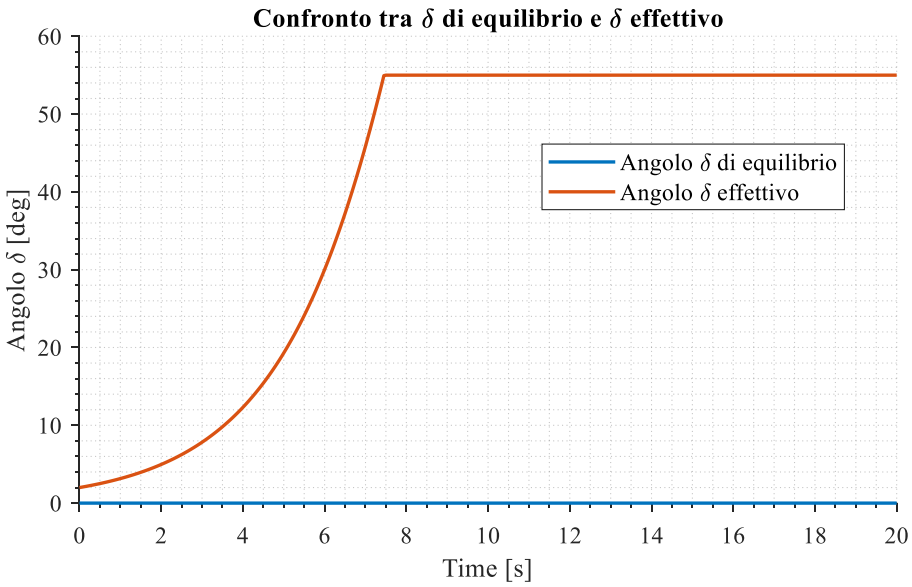

Figura 3.8 – Andamento dell'angolo δ lungo il tratto rettilineo

Nella simulazione in Figura 3.7 si vede che l'evoluzione dell'angolo δ non è controllata. In particolare, si può verificare che, se il robot mobile si sta muovendo in retromarcia lungo una traiettoria prestabilita, l'angolo δ diverge e se il percorso in retromarcia supera una certa

lunghezza si arriverà all'urto tra i due moduli (quindi al manifestarsi del fenomeno del jackknife). In assenza di collisioni, si raggiungerebbe l'angolo di equilibrio $\delta = \pi$.

L'andamento che viene mostrato nelle simulazioni può anche essere dimostrato analiticamente considerando l'approssimazione lineare del modello matematico del sistema trattore-rimorchio attorno alla traiettoria di riferimento e osservando che è instabile a causa della presenza di due autovalori con parte reale positiva o sfruttando una equazione differenziale come modello della posizione del rimorchio rispetto alla motrice a cui è attaccato [37].

Affermare che in marcia indietro l'angolo δ aumenta equivale a dire che la soluzione $\delta_{eq,1} = 0$ rappresenta una condizione di equilibrio instabile per il robot, il quale tende invece a portarsi nella sua condizione di equilibrio stabile in retromarcia, cioè alla soluzione $\delta_{eq,2} = \pi$. La soluzione $\delta_{eq,2} = \pi$, benchè sia la condizione a cui tende naturalmente il robot, corrisponde all'avere modulo anteriore e posteriore sovrapposti, cosa non realizzabile nella realtà in quanto il valore di δ è limitato da arresti meccanici in corrispondenza del giunto (esiste un $\delta_{max} < \pi$). Questo equivale a dire che il robot, se non viene opportunamente controllato durante le fasi in retromarcia, tende sempre al fenomeno del jackknife. L'obiettivo di questo studio diventa perciò quello di evitare che il robot tenda alla sua condizione di equilibrio stabile e cercare, al contrario, di mantenerlo nella condizione instabile $\delta_{eq,1}$ durante il moto in retromarcia lungo un tratto rettilineo.

I risultati trovati e le considerazioni riguardanti il moto rettilineo in retromarcia di Epi.q valgono anche per il rover Agri.q.

L'unico caso in cui, dal punto di vista puramente cinematico, non è necessario l'utilizzo di un controllore, corrisponde alla situazione in cui il robot inizia la traiettoria rettilinea in retromarcia avendo i due moduli già perfettamente allineati. Questo caso è però difficilmente riscontrabile nella realtà in quanto basterebbe una piccola perturbazione (data dal terreno o da forze esterne) per spostare l'angolo δ dal valore nullo e quindi portare alla chiusura del modulo posteriore su quello anteriore dopo un certo intervallo di tempo.

3.2.2 *Traiettoria circolare*

Anche in questo caso, per determinare le condizioni di equilibrio, si considera l'equazione

$$\dot{\delta} = \left(\frac{a}{b}\cos\delta + 1\right)\dot{\varphi}_1 - \frac{1}{b}\sin\delta \cdot v_1$$

Ponendo $\dot{\delta} = 0$ e ricordando che $v_1 = \frac{\dot{\varphi}_1}{\rho_1}$ si ottiene

$$a\cos\delta - \frac{1}{\rho_1}\sin\delta + b = 0 \tag{3.18}$$

Si ottiene un'equazione lineare trigonometrica che è possibile risolvere mediante le formule parametriche. Introducendo la sostituzione

$$t = \tan\left(\frac{\delta}{2}\right) \quad \Rightarrow \quad \begin{cases} \sin\delta = \dfrac{2t}{1+t^2} \\ \cos\delta = \dfrac{1-t^2}{1+t^2} \end{cases}$$

e applicandola all'espressione (3.18) si ottiene l'equazione di secondo grado

$$(b-a)t^2 - \frac{2}{\rho_1}t + (b+a) = 0$$

Risolvendo tale equazione si ricavano due soluzioni possibili

$$t_{1,2} = \frac{\dfrac{1}{\rho_1} \pm \sqrt{\dfrac{1}{{\rho_1}^2} + a^2 - b^2}}{b-a}$$

Quindi

$$\begin{cases} t_1 = \tan\left(\dfrac{\delta_1}{2}\right) = \dfrac{\dfrac{1}{\rho_1} + \sqrt{\dfrac{1}{{\rho_1}^2} + a^2 - b^2}}{b-a} \\ \dfrac{\delta_1}{2} \neq \dfrac{\pi}{2} + k\pi \qquad k \in \mathrm{Z} \end{cases} \quad \Rightarrow \quad \begin{cases} \delta_{eq,1} = 2\,tan^{-1}\left(\dfrac{\dfrac{1}{\rho_1} + \sqrt{\dfrac{1}{\rho_1^2} + a^2 - b^2}}{b-a}\right) \\ \delta_{eq,1} \neq \pi + 2k\pi \qquad k \in Z \end{cases}$$

$$\begin{cases} t_2 = \tan\left(\dfrac{\delta_2}{2}\right) = \dfrac{\dfrac{1}{\rho} - \sqrt{\dfrac{1}{\rho^2} + a^2 - b^2}}{b-a} \\ \dfrac{\delta_2}{2} \neq \dfrac{\pi}{2} + k\pi \qquad k \in \mathrm{Z} \end{cases} \quad \Rightarrow \quad \begin{cases} \delta_{eq,2} = 2\,tan^{-1}\left(\dfrac{\dfrac{1}{\rho_1} - \sqrt{\dfrac{1}{\rho_1^2} + a^2 - b^2}}{b-a}\right) \\ \delta_{eq,2} \neq \pi + 2k\pi \qquad k \in Z \end{cases}$$

Anche in questo caso è possibile effettuare delle simulazioni basandosi sul modello cinematico descritto in precedenza. Si simula perciò il comportamento di un veicolo articolato con le stesse caratteristiche cinematiche del robot Epi.q.

Robot: Epi.q

Tipo di moto: marcia avanti, traiettoria curvilinea con $\rho_1 > 0$
$R_1 = \frac{1}{\rho_1} = 0.91\ m$

Condizioni iniziali:
$\varphi_{1,O} = 0°$
$\delta_O = -30°$

Angolo δ di equilibrio:
$\delta_{eq,1} = 179.5°$
$\delta_{eq,2} = 16.95°$

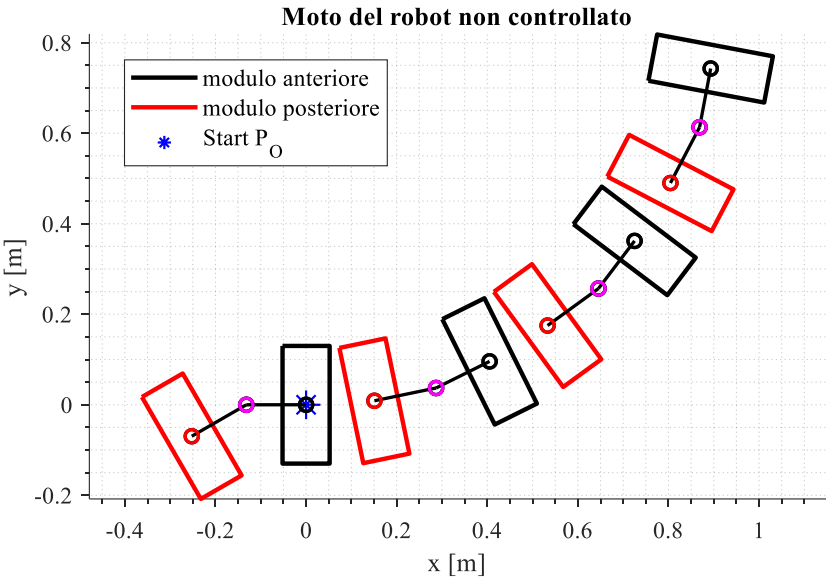

Figura 3.9 – Simulazione del moto in marcia avanti

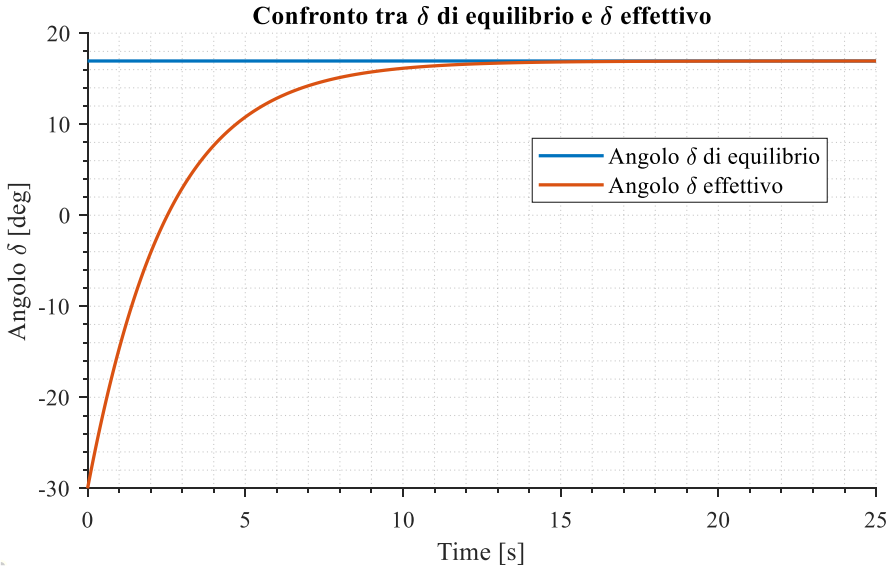

Figura 3.10 – Andamento dell'angolo δ lungo il tratto curvilineo

Robot: Epi.q

Tipo di moto: marcia indietro, traiettoria curvilinea con $\rho_1 > 0$
$R_1 = \frac{1}{\rho_1} = 0.91\ m$

Condizioni iniziali:
$\varphi_{1,O} = 0°$
$\delta_O = 0°$

Angolo δ di equilibrio:
$\delta_{eq,1} = 179.5°$
$\delta_{eq,2} = 16.95°$

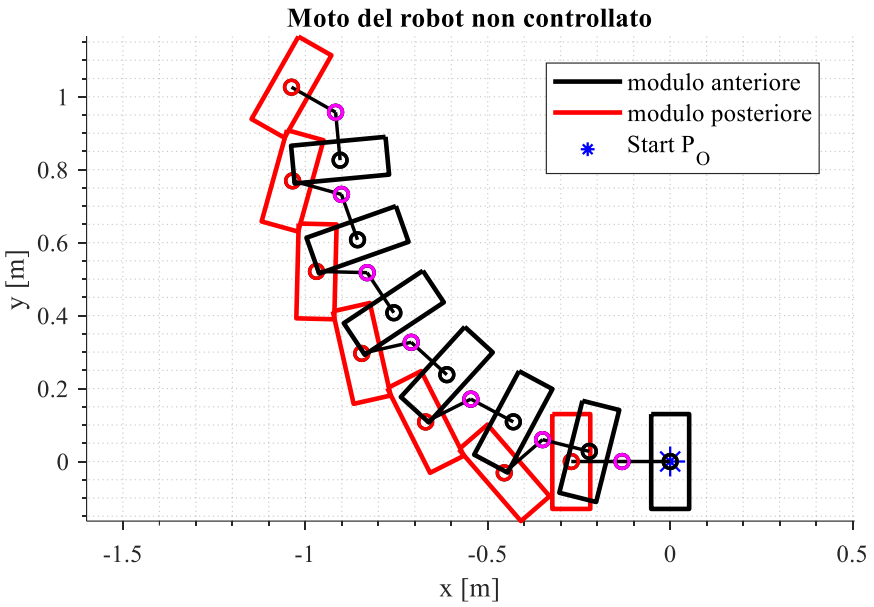

Figura 3.11 – Simulazione del moto in marcia indietro

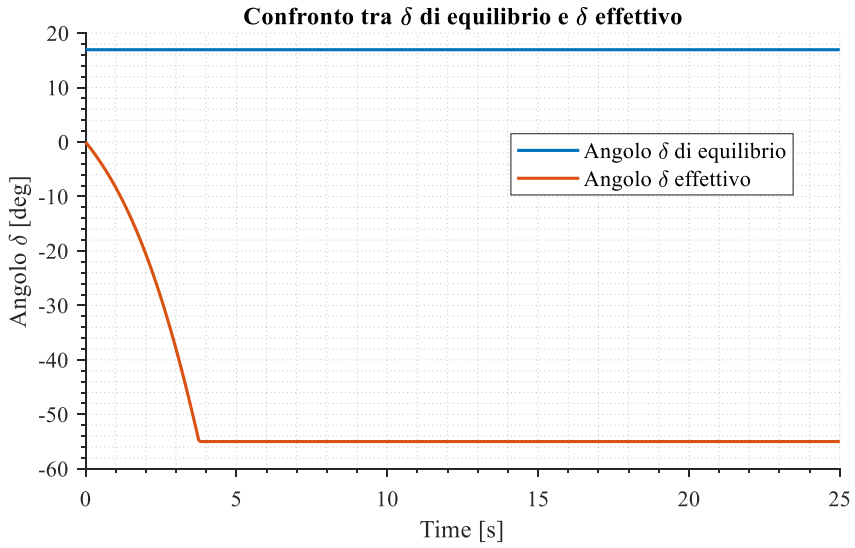

Figura 3.12 – Andamento dell'angolo δ lungo il tratto curvilineo

Da queste simulazioni si può dedurre che, quando la curvatura è positiva, l'angolo di equilibrio di riferimento corrisponde alla soluzione $\delta_{eq,2}$ (mentre $\delta_{eq,1}$ è superiore all'angolo limite δ_{max}). Tuttavia, mentre in marcia avanti $\delta_{eq,2}$ risulta essere la soluzione stabile, in marcia indietro è instabile (in modo analogo a quanto visto per il tratto rettilineo). Quindi, se il robot non viene controllato nelle fasi di retromarcia, esso tende a portarsi nella condizione di equilibrio stabile ($\delta_{eq,1}$); l'angolo δ tende a divergere e, se questo fenomeno si protrae oltre una certa lunghezza del percorso, si arriverà al fenomeno del jackknife.

In caso di raggio di curvatura negativo, facendo le simulazioni si ottiene che la soluzione di riferimento è $\delta_{eq,1}$ (negativa). Anche tale soluzione risulta essere stabile in marcia avanti ed instabile in marcia indietro, quindi in marcia avanti il robot si porta automaticamente nella condizione $\delta_{eq,1}$, mentre in marcia indietro servirà l'intervento di un controllore per poter mantenere il robot in tale condizione senza che vi sia il ripiegamento del modulo posteriore su quello anteriore.

Riassumendo, se il robot va in avanti, l'angolo di equilibrio stabile è quello più vicino a 0, se va in marcia indietro l'angolo di equilibrio stabile è quello più vicino a π. Inoltre, il segno dell'angolo di equilibrio dipende dal segno della curvatura: se la curvatura è positiva δ_{eq} è positivo, se è negativa δ_{eq} è negativo, ciò vale in entrambi i sensi di marcia.

Per cui, anche in caso di traiettoria curvilinea, è necessario elaborare un sistema di controllo che sia in grado di mantenere il robot nella sua condizione di equilibrio instabile in caso di moto in retromarcia. Questo spiega il perché il moto in retromarcia risulti particolarmente problematico per un veicolo articolato.

3.2.3 Il caso di Agri.q

Se si prende in considerazione il rover Agri.q che percorre una traiettoria curvilinea, l'equazione (3.8) che esprime $\dot{\delta}$ in funzione di $\dot{\varphi}_1$ e v_1 può essere semplificata in quanto la dimensione a per questo robot è nulla.

$$\dot{\delta} = \dot{\varphi}_1 - \frac{1}{b}\sin\delta \cdot v_1 = \dot{\varphi}_1\left(1 - \frac{1}{\rho_1 b}\sin\delta\right)$$

Per determinare l'angolo δ di equilibrio poniamo $\dot{\delta} = 0$, come fatto per il robot Epi.q, ottenendo:

$$\sin\delta_{eq} = \rho_1 b$$

Quindi l'angolo δ di equilibrio del robot Agri.q che percorre una traiettoria curvilinea risulta essere:

$$\begin{cases} \delta_{eq} = \sin^{-1}(\rho_1 b) \\ -1 \leq \rho_1 b \leq 1 \end{cases}$$

Anche in questo caso, se il robot si muove in marcia avanti, l'angolo di equilibrio stabile è quello più vicino a 0, se va in marcia indietro l'angolo di equilibrio stabile è quello più vicino

a π. Questo comportamento è visibile nelle seguenti simulazioni, eseguite considerando le caratteristiche cinematiche del robot Agri.q.

Robot: Agri.q

Tipo di moto: marcia avanti, traiettoria curvilinea con $\rho_1 > 0$
$R_1 = \frac{1}{\rho_1} = 5.21\ m$

Condizioni iniziali:
$\varphi_{1,O} = 0°$
$\delta_O = -30°$

Angolo δ di equilibrio:
$\delta_{eq,1} = 14.45°$
$\delta_{eq,2} = 165.55°$

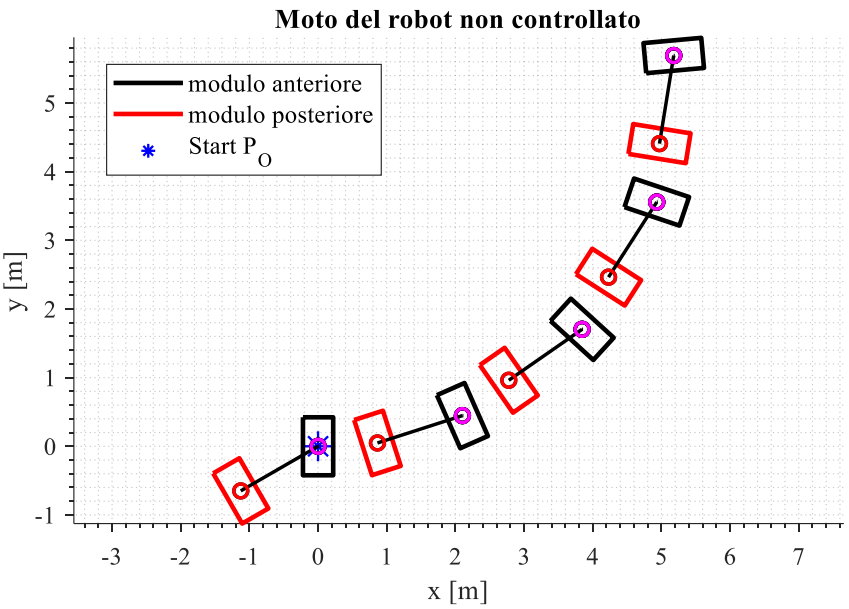

Figura 3.13 – Simulazione del moto in marcia avanti

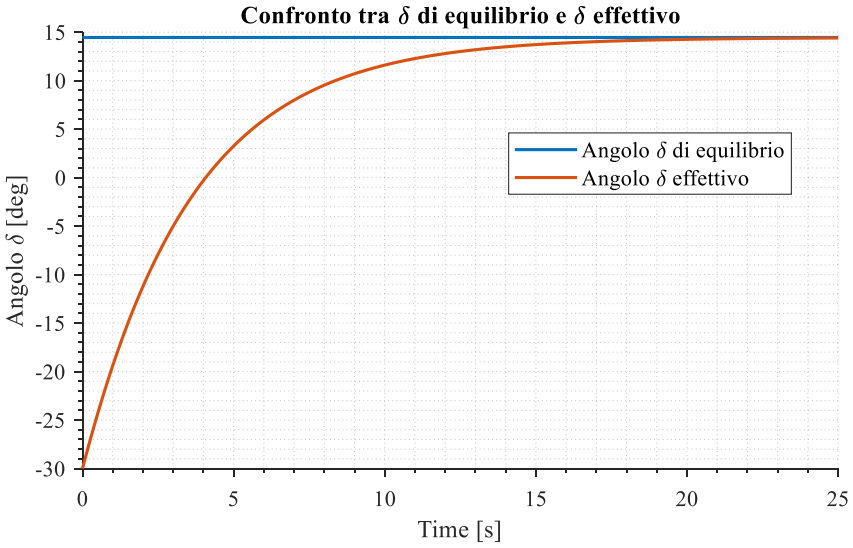

Figura 3.14 – Andamento dell'angolo δ lungo il tratto curvilineo

Robot: Agri.q

Tipo di moto: marcia indietro, traiettoria curvilinea con $\rho_1 > 0$
$R_1 = \frac{1}{\rho_1} = 5.21\ m$

Condizioni iniziali:
$\varphi_{1,O} = 0°$
$\delta_O = 0°$

Angolo δ di equilibrio:
$\delta_{eq,1} = 14.45°$
$\delta_{eq,2} = 165.55°$

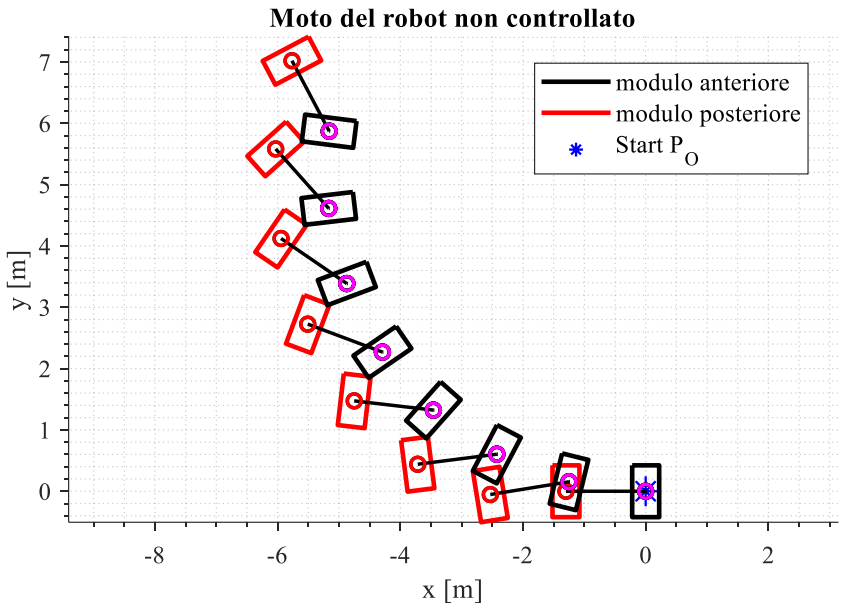

Figura 3.15 – Simulazione del moto in marcia indietro

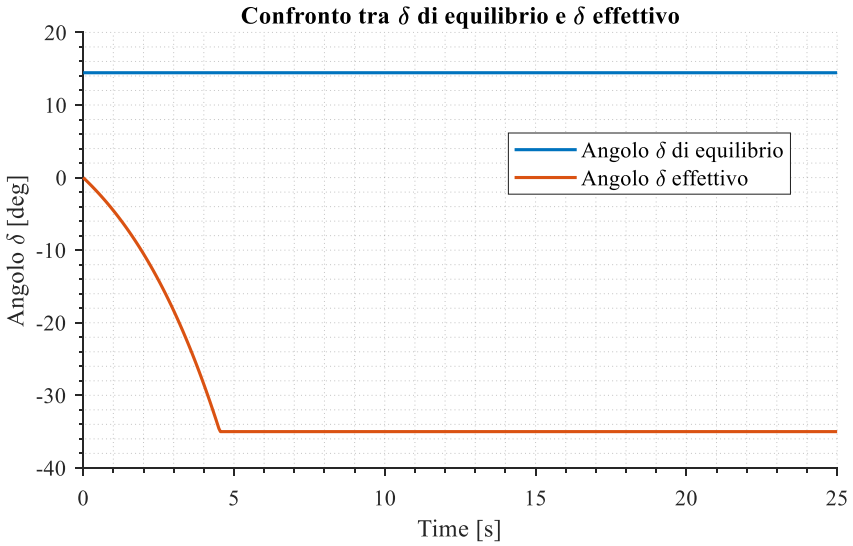

Figura 3.16 – Andamento dell'angolo δ lungo il tratto curvilineo

Le considerazioni fatte per il robot Epi.q valgono anche per Agri.q: in marcia avanti l'angolo δ di equilibrio più vicino a 0 è la soluzione stabile, mentre in marcia indietro risulta essere la soluzione instabile. Quindi, se il robot non viene controllato in marcia indietro, il rimorchio tende a ripiegarsi sull'avantreno poiché l'angolo δ tende a divergere allontanandosi dalla sua condizione di equilibrio instabile.

Al termine di quest'analisi sull'angolo δ possiamo concludere che, sia in caso di traiettoria rettilinea che curvilinea, è necessario elaborare un sistema di controllo che sia in grado di stabilizzare l'angolo di equilibrio in caso di moto in retromarcia.

3.3 Controllore cinematico di basso livello

Come visto nel paragrafo precedente, per comprendere il comportamento del robot in retromarcia, bisogna fare riferimento agli angoli δ_{eq} instabili. A causa di questa instabilità delle condizioni di equilibrio, se il robot non viene controllato, durante le fasi in retromarcia, tende sempre al fenomeno del jackknife, cioè il modulo posteriore tende sempre a chiudersi su quello anteriore. Risulta perciò necessario capire come effettuare il moto in marcia indietro mantenendo il controllo del modulo posteriore.

Un primo approccio per risolvere questo problema consiste nello sviluppo di un controllore di basso livello, cioè un sistema che si occupa di controllare gli attuatori del robot secondo le istruzioni del controllo di alto livello. Il controllo di alto livello decide il movimento, mentre quello di basso livello si occupa di controllare i motori affinché tale movimento sia attuato.

In questo caso il controllore riceve dall'alto livello (un utente in caso di guida manuale o un blocco di pianificazione di traiettoria in caso di guida autonoma) dei riferimenti di velocità da dare alle ruote del modulo anteriore, in quale, per entrambi i robot, viene considerato l'unico dotato di ruote motrici. Il controllore fa in modo di modulare tali velocità, per poi mandarle al modello cinematico del robot, con l'obiettivo di raggiungere e stabilizzare l'angolo δ di equilibrio, il quale viene calcolato per ogni punto del percorso che il robot è chiamato ad eseguire in retromarcia.

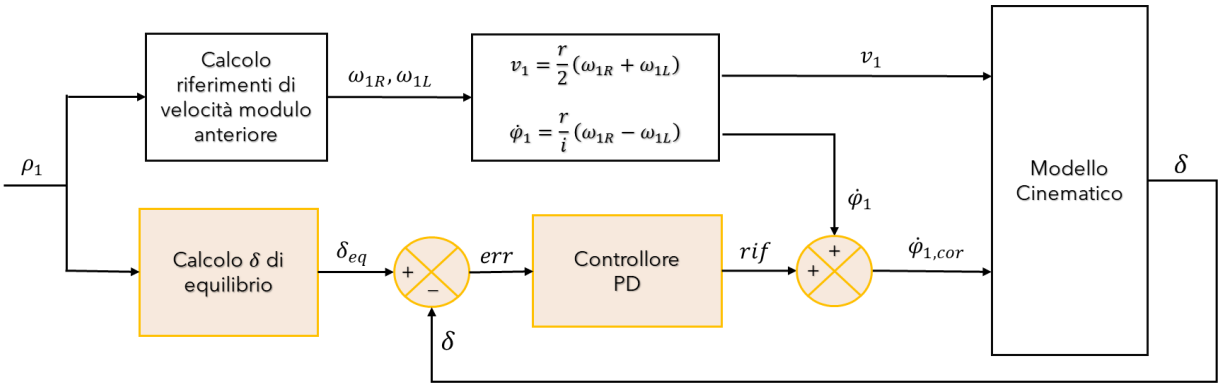

Figura 3.17 – Schema del controllore di basso livello (stabilizzatore dell'angolo δ).
Il controllo di basso livello è costituito dai blocchi in arancione.

Come si può vedere dallo schema in Figura 3.17 il controllore riceve come input la curvatura ρ_1 di riferimento riferita al modulo frontale del robot articolato. Il modo in cui viene definita ρ_1 verrà discusso nei capitoli successivi.

Noto il valore di ρ_1 è possibile calcolare l'angolo δ_{eq} mediante le equazioni descritte nella Sezione 3.2. Tale valore di δ rappresenta l'obiettivo del controllo, ossia l'angolo che si vuole far mantenere al robot in quel determinato istante di moto. Il valore di riferimento δ_{eq} viene confrontato col valore effettivo δ risultante dal modello cinematico: se $\delta \neq \delta_{eq}$ allora vuol dire che il robot nel moto all'indietro non si trova nella condizione voluta ed è necessario applicare una correzione al fine di ridurre il divario tra δ e δ_{eq}. Viene quindi generato un errore err pari alla differenza tra δ_{eq} e δ. Questo parametro viene mandato ad un controllore Proporzionale-Derivativo (PD) in modo da ottenere la stabilizzazione dell'angolo δ:

$$rif = K_P \cdot err + K_D \cdot \frac{d(err)}{dt} \tag{3.19}$$

$$rif = K_P(\delta_{eq} - \delta) + K_D \cdot \frac{d(\delta_{eq} - \delta)}{dt} \tag{3.20}$$

dove K_P e K_D sono i guadagni proporzionale e derivativo.

Il valore rif così ottenuto viene quindi sommato al valore della velocità angolare di riferimento $\dot{\varphi}_1$. Da questa somma si ottiene il valore della velocità angolare corretta $\dot{\varphi}_{1,\mathrm{cor}}$ che deve essere data in input al modello cinematico del robot al fine di seguire il δ_{eq}.

3.3.1 Tuning del controllore PD

L'utilizzo del controllore PD richiede l'ottimizzazione dei guadagni K_P e K_D. Per svolgere questo compito è possibile ricorrere ad una tecnica di ottimizzazione standard: si parte ponendo $K_P = 1$ e $K_D = 0$, inizializzando il sistema con un angolo δ diverso da zero e controllandolo a zero durante la retromarcia a velocità costante. Successivamente K_P viene aumentato fino a quando appare l'oscillazione. Una volta trovato il valore di K_P ottimale, si passa ad aumentare K_D. Introducendo il guadagno derivativo (tarato in modo opportuno) è possibile attenuare le oscillazioni che il controllo stesso causa sull'andamento della velocità angolare $\dot{\varphi}_1$. Attraverso questa tecnica sono stati trovati i valori ottimali: $K_P = 4$ e $K_D = 0.015$.

3.3.2 Perché applicare la correzione alla velocità angolare

Come si può vedere nello schema in Figura 3.17, il controllore interviene solo sulla velocità angolare $\dot{\varphi}_1$. Tuttavia, la velocità $\dot{\delta}$ dipende sia da $\dot{\varphi}_1$ che da v_1, aspetto chiaramente visibile dall'equazione:

$$\dot{\delta} = \left(\frac{a}{b}\cos\delta + 1\right)\dot{\varphi}_1 - \frac{1}{b}\sin\delta \cdot v_1$$

Già da questa relazione si può intuire che per $\delta = 0$ l'unico termine che interviene è quello che moltiplica $\dot{\varphi}_1$, mentre per $\delta = 90°$ (valore non raggiungibile nella realtà in quanto maggiore di δ_{max}) il contributo maggiore viene dato da v_1. Quindi, a seconda del valore dell'angolo δ assunto dal robot in quell'istante, si potrebbe avere un maggior effetto del controllo se applicato alla velocità longitudinale o angolare.

Risulta pertanto interessante eseguire un'analisi di sensibilità su tale equazione al fine di determinare un δ discriminante (δ_D) al di sotto del quale, dal punto divista del controllo, ha più effetto l'azione su $\dot{\varphi}_1$ e al di sopra del quale ha più effetto l'azione su v_1. Il δ discriminante corrisponde a quel valore di δ tale per cui il contributo dato da $\dot{\varphi}_1$ uguaglia quello dato da v_1. Quindi si vuole determinare quel valore di δ tale per cui, nell'espressione (3.8), il termine che moltiplica la velocità angolare sia uguale al termine che moltiplica la velocità longitudinale. Ciò significa che il δ discriminante è quell'angolo che soddisfa l'equazione:

$$-\frac{1}{b}\sin\delta_D = \frac{a}{b}\cos\delta_D + 1$$

$$a\cos\delta_D + \sin\delta_D + b = 0$$

Poiché si tratta di una equazione trigonometrica si ricorre nuovamente alle formule parametriche ottenendo:

$$(b-a)t^2 - 2t + (b+a) = 0 \qquad \text{dove} \qquad t = \tan\left(\frac{\delta_D}{2}\right)$$

Risolvendo l'equazione trigonometrica si ottengono due soluzioni possibili:

$$\begin{cases} t_1 = \tan\left(\dfrac{\delta_{D,1}}{2}\right) = \dfrac{-1+\sqrt{1+a^2-b^2}}{b-a} \\ \dfrac{\delta_{D,1}}{2} \neq \dfrac{\pi}{2} + k\pi \qquad k \in Z \end{cases} \Rightarrow \begin{cases} \delta_{D,1} = 2\tan^{-1}\left(\dfrac{-1+\sqrt{1+a^2-b^2}}{b-a}\right) \\ \delta_{D,1} \neq \pi + 2k\pi \qquad k \in Z \end{cases}$$

$$\begin{cases} t_2 = tan\left(\dfrac{\delta_{D,2}}{2}\right) = \dfrac{-1-\sqrt{1+a^2-b^2}}{b-a} \\ \dfrac{\delta_{D,2}}{2} \neq \dfrac{\pi}{2} + k\pi \qquad k \in Z \end{cases} \Rightarrow \begin{cases} \delta_{D,2} = 2\tan^{-1}\left(\dfrac{-1-\sqrt{1+a^2-b^2}}{b-a}\right) \\ \delta_{D,2} \neq \pi + 2k\pi \qquad k \in Z \end{cases}$$

Per quanto riguarda il robot Epi.q, note le dimensioni a e b del robot si ricava:

$$\delta_{D,1} = -15.4°$$
$$\delta_{D,2} = -179.6°$$

La soluzione $\delta_{D,2}$ non può essere tenuta in considerazione in quanto supera i limiti imposti da δ_{max}.

In conclusione, poiché il comportamento di $\dot{\delta}$ è simmetrico rispetto all'asse $\delta = 0$, per il robot Epi.q si può affermare che:

- Se $-\delta_{max} < \delta < -15.4° \; oppure \; 15.4° < \delta < \delta_{max}$ allora l'azione del controllo sulla velocità v_1 è maggiore dell'azione su $\dot{\varphi}_1$.
- Se $-15.4° \leq \delta \leq 15.4°$ allora l'azione del controllo sulla velocità $\dot{\varphi}_1$ è maggiore dell'azione su v_1.

Per quanto riguarda il robot Agri.q è necessario fare delle considerazioni specifiche per questo robot. Poiché Agri.q è caratterizzato dall'avere la dimensione $a = 0$ l'equazione (3.8) diventa:

$$\dot{\delta} = \dot{\varphi}_1 - \frac{1}{b}\sin\delta \cdot v_1$$

Quindi il δ discriminante è quell'angolo che soddisfa l'equazione:

$$-\frac{1}{b}\sin\delta_D = 1 \quad \Rightarrow \quad \sin\delta_D = -b$$

Ma, poiché $b > 1$ e $-1 \leq \sin\delta_D \leq 1$, non esiste nessun valore di δ tale per cui l'effetto dato da $\dot{\varphi}_1$ uguagli quello dato da v_1 in termini di controllo del modulo posteriore. In particolare, si nota che:

$$\frac{1}{b}\sin\delta < 1 \text{ sempre}$$

Quindi, quando si considera il robot Agri.q, il contributo di $\dot{\varphi}_1$ è sempre maggiore di quello dato da v_1, perciò per tale tipologia di robot è consigliabile che il controllo vada ad agire sempre sulla velocità angolare.

Queste stesse conclusioni possono essere ricavate anche dall'analisi dei modelli dei due robot. Dalle rappresentazioni in Figura 3.18 e 3.19 si può vedere che la grandezza che controlla l'andamento del modulo posteriore è la direzione della velocità in corrispondenza del giunto J. Poiché su Agri.q l'elemento di dimensione a è nullo, l'unico effetto che può controllare l'orientazione del modulo posteriore è la rotazione del modulo anteriore, ciò giustifica il fatto che solo la velocità angolare gestisca il controllo. Invece con Epi.q, essendo presente il braccio a, è possibile controllare l'andamento del modulo posteriore agendo anche solo sulla velocità longitudinale (senza sterzare l'avantreno).

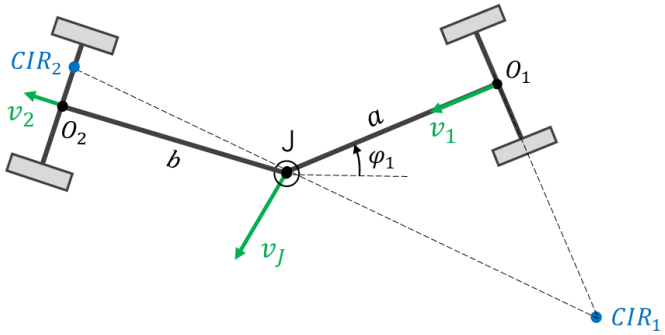

Figura 3.18 – Modello robot Epi.q

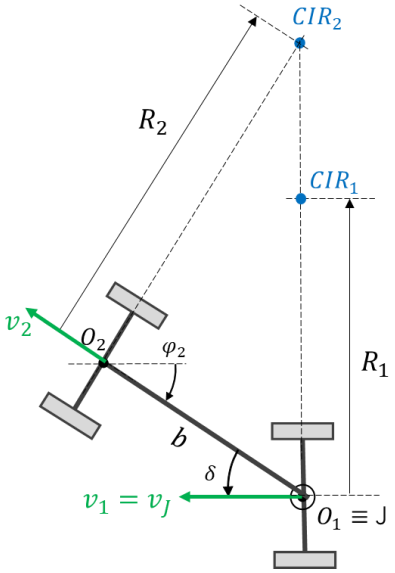

Figura 3.19 – Modello robot Agri.q

Per le considerazioni sopra riportate e per cercare di ottenere un controllo semplice e valido per entrambi i robot sotto esame è stata adottata una strategia di controllo che vada ad agire sempre sulla velocità angolare $\dot{\varphi}_1$.

3.3.3 Risultati ottenuti mediante il controllo

Vengono proposti alcuni esempi.

Robot: Epi.q

Tipo di moto: marcia indietro, traiettoria rettilinea

Condizioni iniziali
$\varphi_{1,0} = 0°$
$\delta_O = 15°$

Angolo δ di equilibrio: $\delta_{eq} = 0°$

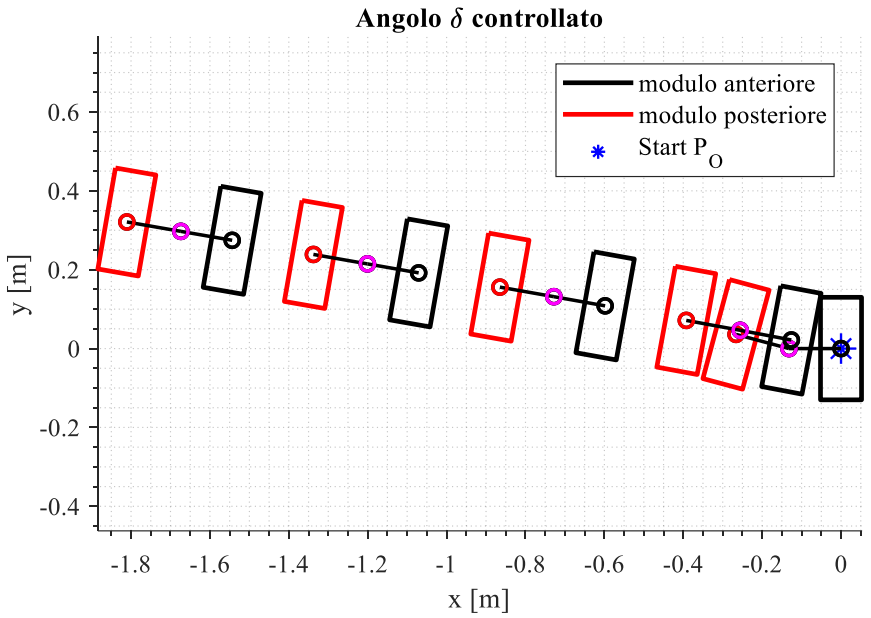

Figura 3.20 – Simulazione del moto in marcia indietro con intervento del controllo sull'angolo δ

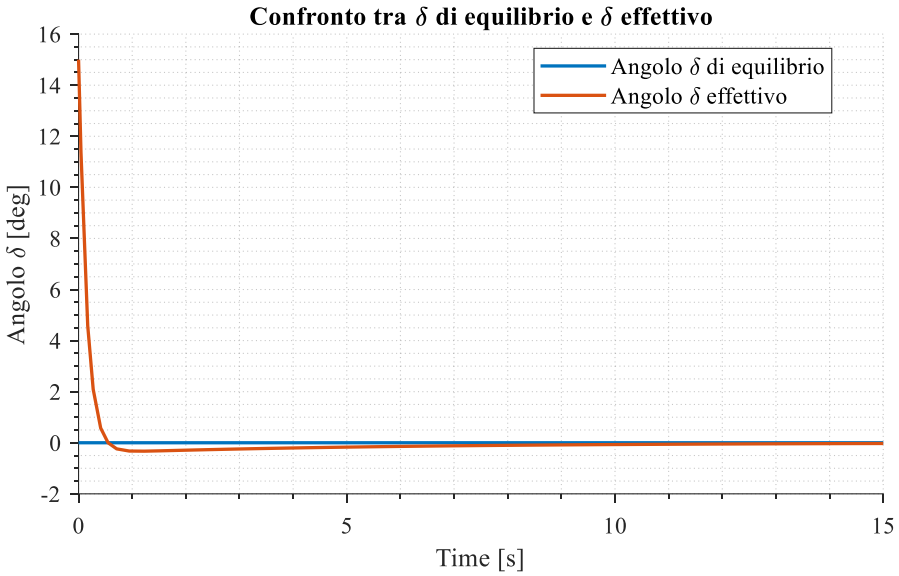

Figura 3.21 – Andamento dell'angolo δ controllato lungo il tratto rettilineo

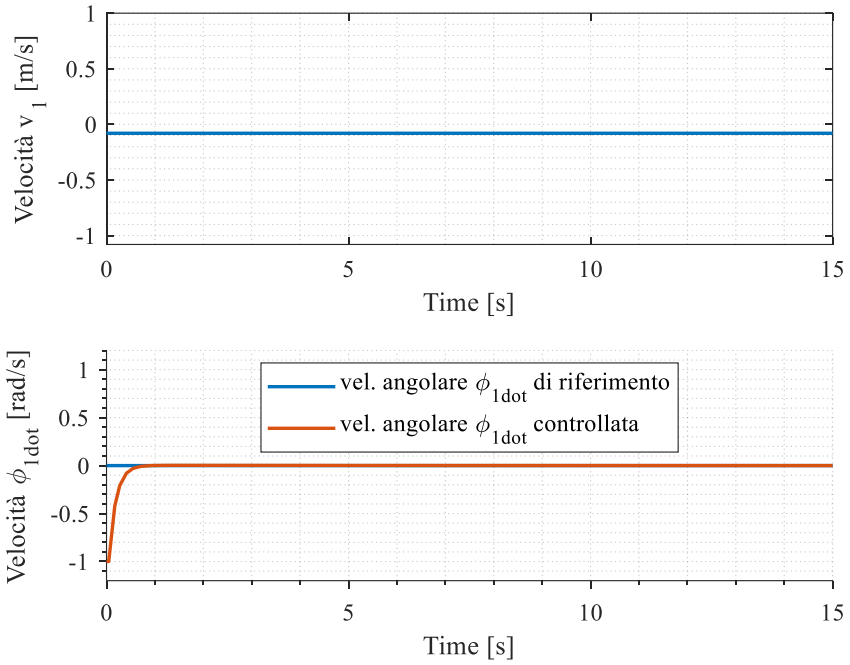

Figura 3.22 – Andamento della velocità longitudinale v_1 e della velocità angolare $\dot{\varphi}_1$

Robot: Epi.q

Tipo di moto: marcia indietro, traiettoria curvilinea con $R_1 = 2.47\ m$

Condizioni iniziali:
$\varphi_{1,O} = 10°$
$\delta_O = -20°$

Angolo δ di equilibrio:
$\delta_{eq} = 6.28°$

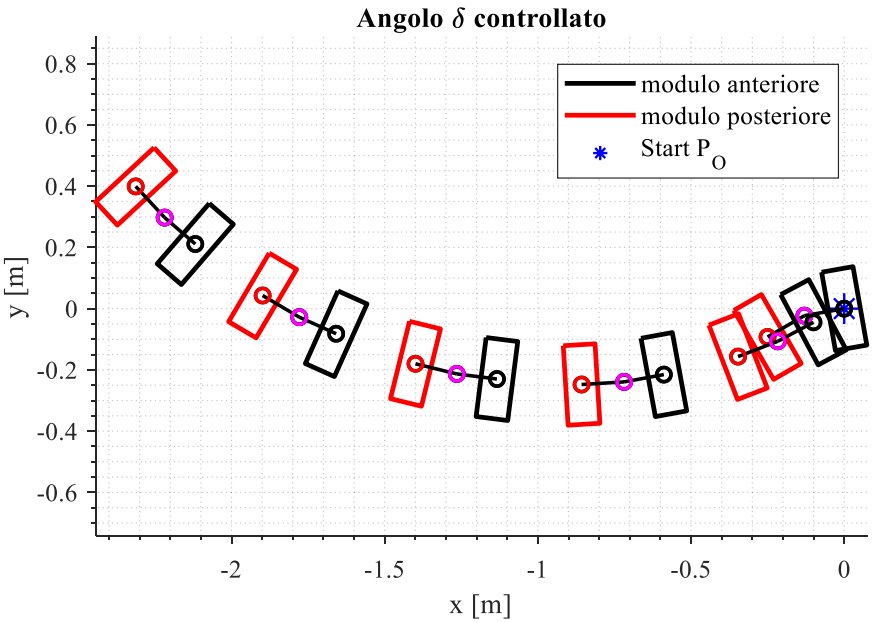

Figura 3.23 – Simulazione del moto in marcia indietro con intervento del controllo sull'angolo δ

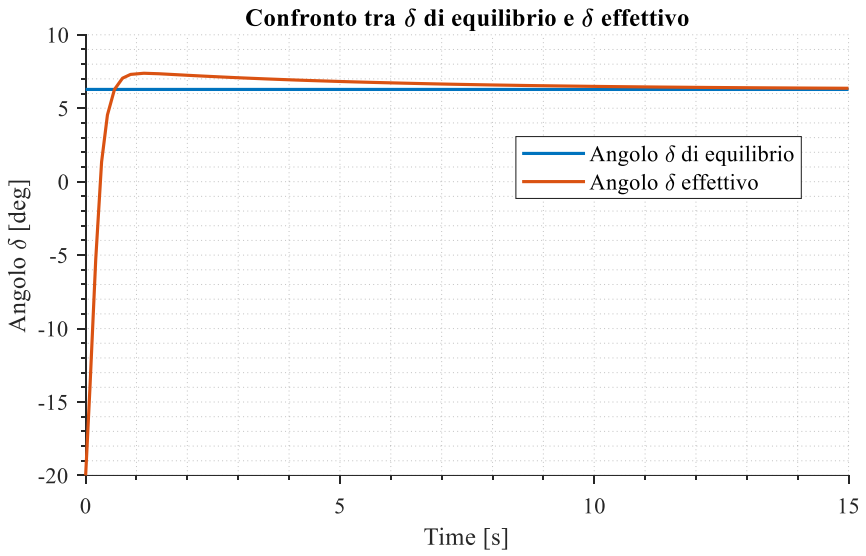

Figura 3.24 – Andamento dell'angolo δ controllato lungo il tratto curvilineo

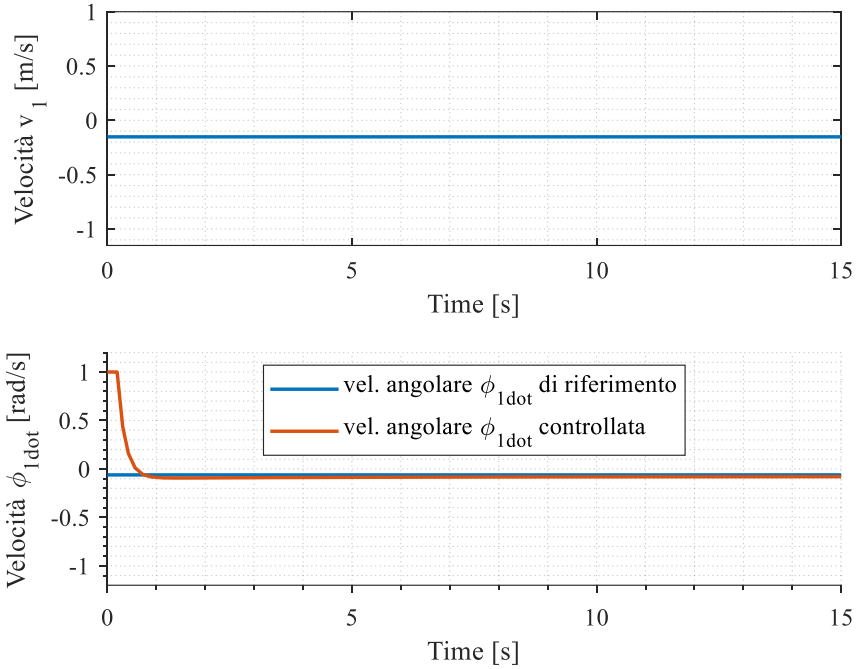

Figura 3.25 – Andamento della velocità longitudinale v_1 e della velocità angolare $\dot{\phi}_1$

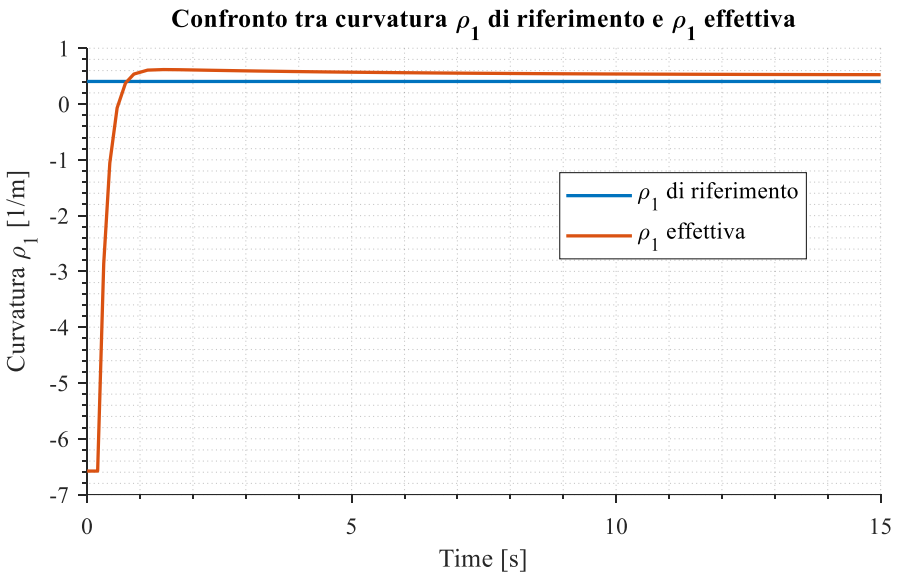

Figura 3.26 – Andamento della curvatura ρ_1

Come si può notare da questa simulazione l'intervento del controllo sulla velocità angolare (passando da un valore di riferimento $\dot{\varphi}_1 = -0.07\,\text{rad/s}$ ad un valore corretto $\dot{\varphi}_{1,cor} = -0.08\,\text{rad/s}$) comporta un conseguente cambiamento della curvatura ρ_1 (e quindi del raggio di curvatura R_1), che si discosta leggermente da quella voluta originariamente (passando da $\rho_1 = 0.2\,m^{-1}$ a $\rho_{1,cor} = 0.28\,m^{-1}$). Ciò significa che il controllore, agendo in modo da stabilizzare l'angolo δ, porta il robot ad allontanarsi dai riferimenti di traiettoria desiderati. Questo è un elemento che va tenuto in considerazione ed è il motivo per il quale, nei capitoli successivi, verrà introdotto un ulteriore controllo con il compito di mantenere il robot sul percorso voluto.

Robot: Agri.q

Tipo di moto: marcia indietro, traiettoria curvilinea con $R_1 = -8\,m$

Condizioni iniziali:
$\varphi_{1,O} = -5°$
$\delta_O = 15°$

Angolo δ di equilibrio:
$\delta_{eq} = -9.32°$

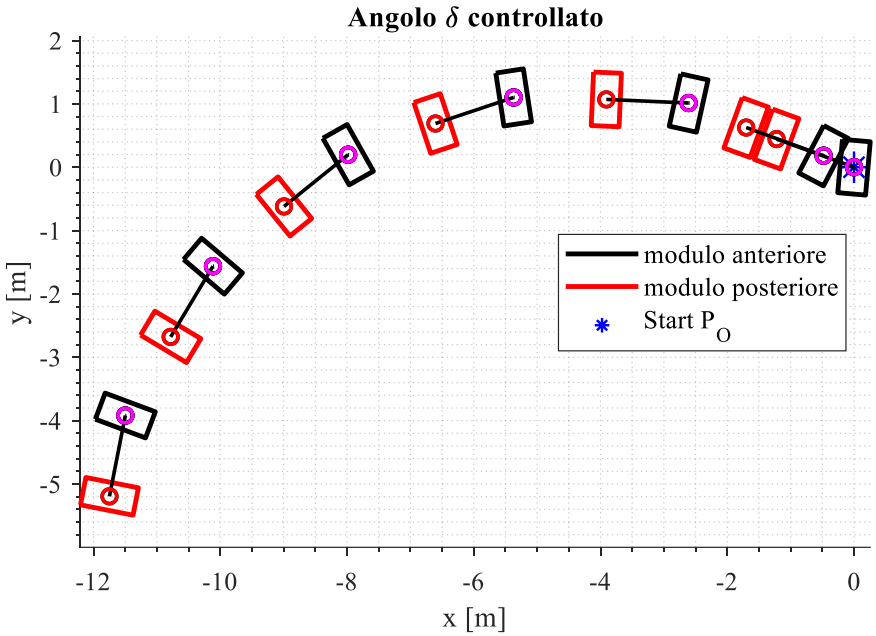

Figura 3.27 – Simulazione del moto in marcia indietro con intervento del controllo sull'angolo δ

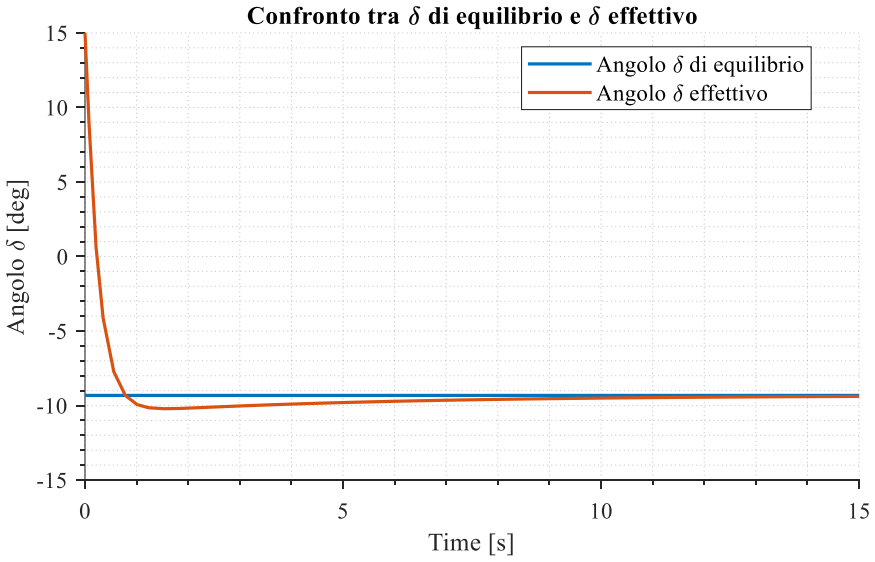

Figura 3.28 – Andamento dell'angolo δ controllato lungo il tratto curvilineo

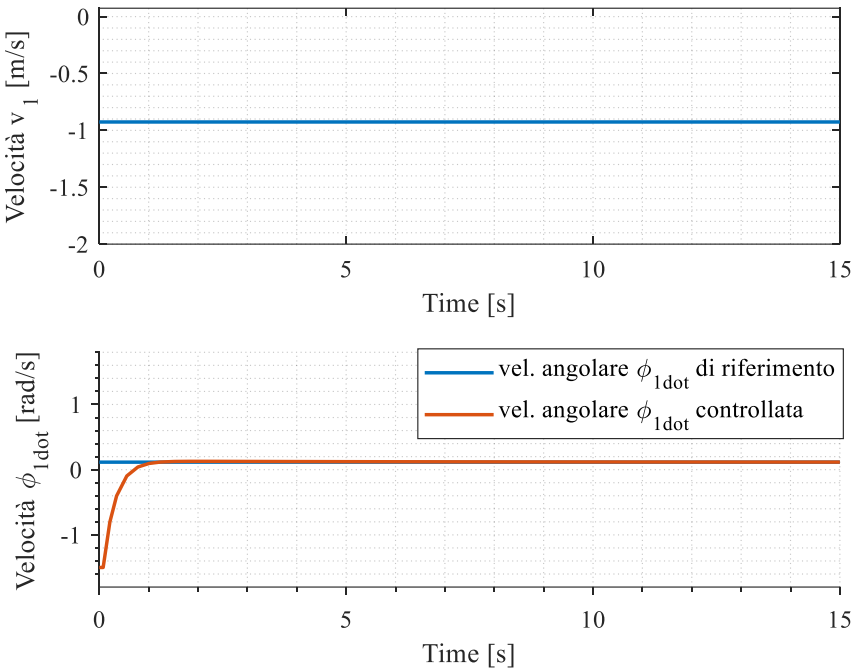

Figura 3.29 – Andamento della velocità longitudinale v_1 e della velocità angolare $\dot{\varphi}_1$

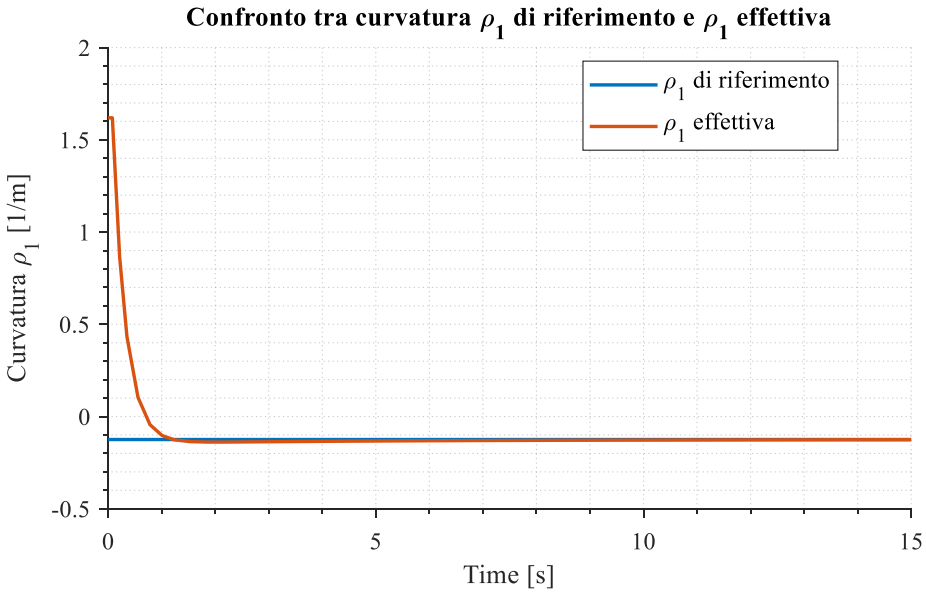

Figura 3.30 – Andamento della curvatura ρ_1

Anche quando si prende in considerazione il robot Agri.q l'intervento del controllo sull'hitch angle porta una variazione della velocità angolare ($\dot{\varphi}_1 = 0.11$ rad/s, $\dot{\varphi}_{1,cor} = 0.12$ rad/s) e quindi della curvatura ($\rho_1 = -0.12\ m^{-1}$, $\rho_{1,cor} = -0.13\ m^{-1}$). Di conseguenza si ha uno scostamento tra la traiettoria desiderata e quella effettivamente seguita dal robot.

4. Capitolo 4: Analisi cinematica basata sui centri di istantanea rotazione

Il modello cinematico di un robot articolato può essere anche espresso in un altro modo. Come già detto nel capitolo precedente, le uniche ruote motrici del robot sono quelle anteriori; quindi, gli unici dispositivi controllabili direttamente sono i motori elettrici che le azionano. Da qui scaturisce l'obiettivo di riuscire a controllare il movimento del modulo anteriore e del modulo posteriore regolando solamente le velocità delle ruote anteriori. Ciò equivale a voler trovare una relazione tra le velocità imposte alle ruote con le posizioni che si vogliono far assumere al robot.

A questo proposito, visto che Epi.q ed Agri.q sono dei robot mobili articolati, un modo alternativo per descrivere questo tipo di robot dal punto di vista cinematico consiste nell'analizzare separatamente i due corpi rigidi che li compongono. Tale approccio è stato adottato anche da Gherlone nella sua tesi [19].

Inizialmente viene trattata l'analisi del modulo frontale, successivamente si analizza il modulo posteriore.

4.1 Modulo anteriore

Come già detto, il modulo anteriore viene identificato con il punto O_1 e viene considerato come l'unico dotato di ruote motrici, dove ω_{1L} e ω_{1R} sono le velocità angolari rispettivamente della ruota sinistra e destra del modulo (anche in questo modello cinematico vale l'ipotesi di considerare un unico assale virtuale per ogni modulo). Quindi, posto che le uniche velocità note inizialmente siano proprio le velocità angolari applicate a ciascuna ruota, si vogliono ricavare le pose assunte dall'avantreno (x_1, y_1, φ_1) nel tempo.

Considerando lo schema del modulo frontale in un istante generico del moto. Noto il raggio r delle ruote, valgono le relazioni:

$$v_{1L} = \omega_{1L} \cdot r$$

$$v_{1R} = \omega_{1R} \cdot r$$

Nonché le relazioni:

$$\dot{\varphi}_1 = \frac{1}{i}(v_{1R} - v_{1L})$$

$$v_1 = \frac{1}{2}(v_{1R} + v_{1L})$$

Note le leggi di velocità in termini di andamento delle velocità angolari applicate alle ruote; essendo il carrello un corpo rigido e conoscendo le velocità in due punti distinti, si può ricavare la velocità di qualsiasi punto del corpo utilizzando il centro di istantanea rotazione CIR_1 già definito nella Sezione 3.1. Noto CIR_1, sapendo che la velocità di un punto è perpendicolare alla retta congiungente il punto con CIR_1, si ricavano le velocità v_1 e v_J (Figura 4.1).

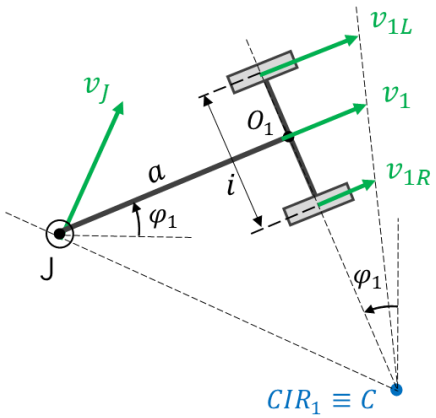

Figura 4.1 – Modello cinematico del modulo frontale del robot Epi.q

Integrando l'espressione della velocità angolare $\dot{\varphi}_1$ è possibile determinare l'angolo φ_1:

$$\varphi_1 = \int \frac{1}{i}(v_{1R} - v_{1L})\, dt \tag{4.1}$$

Conoscendo φ_1 si possono calcolare le posizioni dei punti O_1 e J:

$$\begin{cases} v_{1x} = v_1 \cos\varphi_1 = \dfrac{1}{2}(v_{1R} + v_{1L}) \cos \displaystyle\int \frac{1}{i}(v_{1R} - v_{1L})\, dt \\ v_{1y} = v_1 \cos\varphi_1 = \dfrac{1}{2}(v_{1R} + v_{1L}) \sin \displaystyle\int \frac{1}{i}(v_{1R} - v_{1L})\, dt \end{cases}$$

$$\begin{cases} x_1 = \int v_{1x}\, dt \\ y_1 = \int v_{1y}\, dt \end{cases} \quad \Rightarrow \quad \begin{cases} x_1 = \displaystyle\int \frac{1}{2}(v_{1R} + v_{1L}) \cos\left(\int \frac{1}{i}(v_{1R} - v_{1L})\, dt\right) dt \\ y_1 = \displaystyle\int \frac{1}{2}(v_{1R} + v_{1L}) \sin\left(\int \frac{1}{i}(v_{1R} - v_{1L})\, dt\right) dt \end{cases}$$

$$\begin{cases} x_J = x_1 - a\cos\varphi_1 \\ y_J = y_1 - a\sin\varphi_1 \end{cases} \tag{4.2}$$

Queste considerazioni valgono anche per il robot Agri.q, l'unica differenza sta nel fatto che in questo robot il termine a è nullo, perciò la cerniera J risulta coincidente con il punto O_1.

4.2 Modulo posteriore

Per lo studio del modulo posteriore viene posta l'attenzione sulla cerniera J, poiché tale elemento è quello di congiunzione tra il modulo anteriore ed il modulo posteriore. Quindi, note la velocità e la posizione del giunto J, l'obiettivo diventa quello di ricavare la posizione e l'orientamento (x_2, y_2, φ_2) del modulo posteriore.

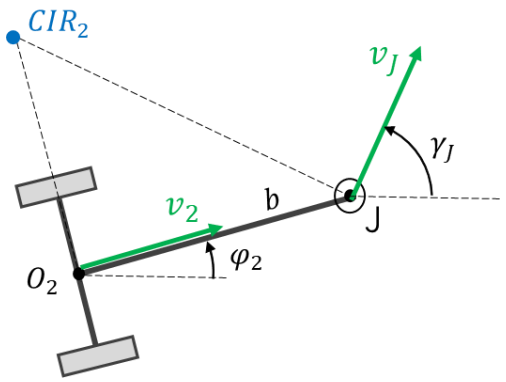

Figura 4.2 – Modello cinematico del modulo posteriore valido per Epi.q ed Agri.q

Conoscendo la velocità v_J è possibile trovare il centro di istantanea rotazione CIR_2 riferito al modulo posteriore, il quale corrisponde al punto di intersezione tra la perpendicolare al vettore velocità v_J ed il prolungamento dell'assale posteriore (come si può vedere nella Figura 4.2). Nota v_J è anche possibile determinare il vettore velocità v_2: tale vettore, infatti, può essere visto come la somma della velocità in J e della velocità che il punto 2 ha rispetto a J, come mostrato nella Figura 4.3.

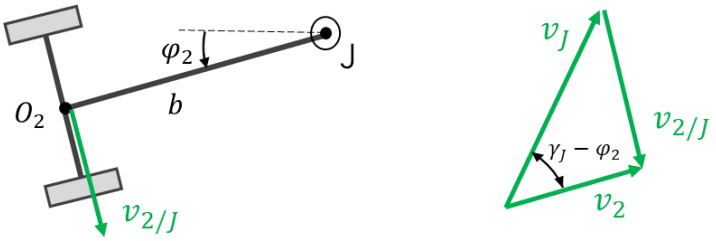

Figura 4.3 –Scomposizione delle velocità del modulo posteriore

$$\bar{v}_2 = \bar{v}_J + \bar{v}_{2/J}$$

$$v_{2/J} = \dot{\varphi}_2 \cdot \overline{JO_2} = \dot{\varphi}_2 \cdot b$$

Dal triangolo di vettori si ricava che:

$$v_2 = v_J \cos(\varphi_2 - \gamma_J)$$

$$v_{2/J} = \dot{\varphi}_2 \cdot b = v_J \sin(\gamma_J - \varphi_2)$$

$$\dot{\varphi}_2 = \frac{v_J \sin(\gamma_J - \varphi_2)}{b} \tag{4.3}$$

$$\varphi_2 = \int \dot{\varphi}_2 \, dt \tag{4.4}$$

Noto l'angolo φ_2, si possono calcolare le coordinate del punto 0_2:

$$\begin{cases} x_2 = x_J - b \cos \varphi_2 \\ y_2 = y_J - b \sin \varphi_2 \end{cases} \tag{4.5}$$

4.3 Approccio ai raggi di curvatura in caso di moto in retromarcia

Si vuole realizzare un moto in retromarcia senza perdere il controllo del modulo posteriore, il quale in caso di moto all'indietro viene spinto e non più trainato, cioè si vuole evitare che si abbia un progressivo aumento in valore assoluto dell'angolo δ dovuto alla tendenza del sistema di portarsi nella condizione di equilibrio stabile.

Per risolvere questo problema e allo stesso tempo ottenere una pianificazione di un percorso efficace è necessario non ragionare più dal punto di vista del modulo fontale (punto O_1) e del raggio di curvatura R_1, ma dal punto di vista del modulo posteriore (punto O_2) e del raggio di curvatura R_2. Tuttavia, il moto del robot non può essere controllato direttamente dalle ruote del modulo posteriore, in quanto non sono motrici.

È però possibile, noto il raggio R_2 del percorso riferito al retrotreno, risalire al corrispondente raggio R_1 del modulo anteriore: in ogni istante di moto il modulo frontale è caratterizzato da un centro di istantanea rotazione CIR_1 e da un raggio di curvatura R_1, mentre al modulo posteriore è possibile associare un centro di istantanea rotazione CIR_2 ed un raggio di curvatura R_2. Nella maggior parte dei casi questi raggi differiscono per modulo anteriore e posteriore, ma, come dimostrato in [19], è possibile correlarli cinematicamente. Quindi, noto R_2 dalla pianificazione del percorso ottimale, è possibile ricavare R_1 tramite una funzione cinematica.

Analizzando la geometria e la cinematica del robot Epi.q si può ricavare l'espressione che lega tra loro i raggi di curvature dei due moduli.

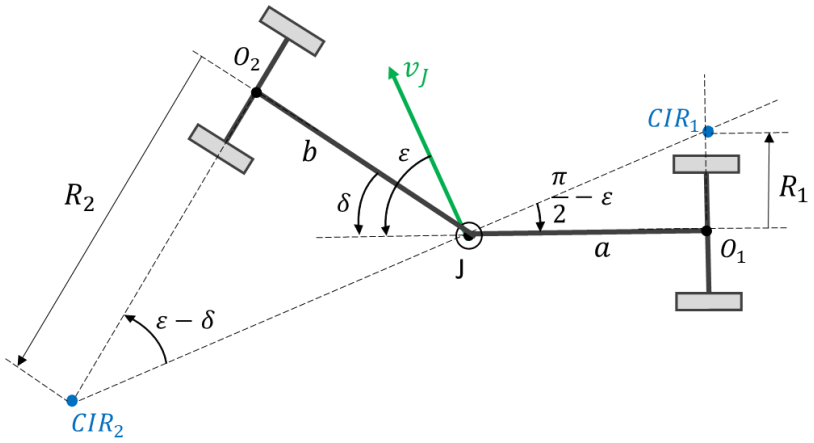

Figura 4.4 – Modello cinematico del robot Epi.q

Osservando la Figura 4.4 è possibile definire:

$$R_1 = a \cdot \tan\left(\frac{\pi}{2} - \varepsilon\right)$$

$$R_2 = \frac{b}{\tan(\delta - \varepsilon)}$$

Dove ε rappresenta l'angolo compreso tra il vettore v_J ed il segmento JO_1. Esplicitando ε dall'espressione di R_2 si ottiene:

$$\varepsilon = \delta - arctan\frac{b}{R_2}$$

Sostituendo ε nell'espressione di R_1 si ricava la funzione cercata:

$$R_1 = a \cdot tan\left(\frac{\pi}{2} - \delta + arctan\frac{b}{R_2}\right) \tag{4.6}$$

Quindi, per passare da R_2 a R_1 basta conoscere la geometria del robot (a e b) e l'angolo di rotazione δ del modulo posteriore. Si ottiene in questo modo un controllo indiretto del retrotreno, il quale viene guidato dal modulo anteriore sfruttando la catena cinematica del robot.

La relazione (4.6) non fornisce però alcuna informazione sulla direzione del moto del modulo frontale, il quale può muoversi in avanti o all'indietro. Infatti, nel moto in retromarcia il moto del modulo posteriore è all'indietro, mentre la direzione del moto del

modulo anteriore non è detto che sia all'indietro. Quindi bisogna determinare in quali situazioni il modulo frontale si muove in avanti ed in quali si muove all'indietro.

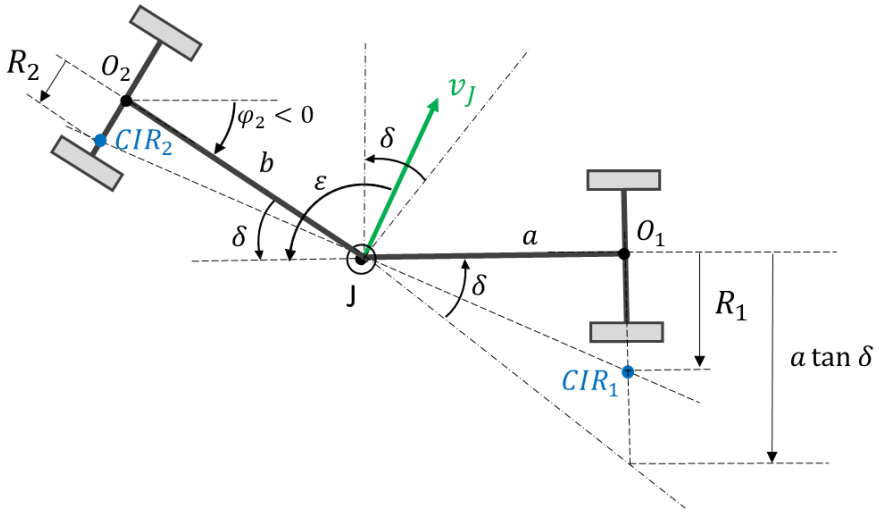

Figura 4.5 – Caso di moto all'indietro con movimento del modulo anteriore in avanti, δ positivo

In Figura 4.5 è rappresentato un caso in cui il modulo anteriore si muove in marcia avanti. In tal caso il vettore velocità v_J ha la componente longitudinale diretta in avanti, ciò implica una velocità positiva del punto O_1 e quindi del modulo frontale. Ciò accade ogniqualvolta è soddisfatta la condizione:

$$|\varepsilon| \geq \frac{\pi}{2}$$

ossia quando, nel caso δ di positivo, ε risulta compreso fra due limiti:

$$\frac{\pi}{2} \leq \varepsilon < \frac{\pi}{2} + \delta$$

Essendo però l'angolo ε non direttamente noto, si ragiona sul raggio di curvatura R_1 del modulo anteriore. All'intervallo dell'angolo ε corrisponde l'intervallo di R_1:

$$-a \tan \delta < R_1 \leq 0 \quad per \quad \delta > 0$$

Il segno negativo è dovuto alla convenzione per cui R_1 è negativo se il robot gira in senso antiorario in marcia indietro. Estendendo la validità a tutto il campo di δ, gli intervalli di valori per i quali il carrello si muove in avanti nonostante si ragioni sul rimorchio sono:

$$\begin{cases} se\ |\delta| \leq \dfrac{\pi}{2} \Rightarrow\ -segn(\delta) a \tan \delta < segn(\delta) R_1 \leq 0 \\ se\ |\delta| > \dfrac{\pi}{2} \Rightarrow\ segn(\delta) R_1 > -segn(\delta) a \tan \delta \quad o \quad segn(\delta) R_1 \leq 0 \end{cases} \tag{4.7}$$

4.3.1 Il caso di Agri,q

Se si prende in considerazione il robot Agri.q l'espressione (4.6) non è utilizzabile, perché, essendo $a = 0$, per ogni valore di R_2 si otterrebbe sempre un raggio di curvatura R_1 nullo. Ciò non corrisponde al comportamento reale del robot in quanto il modulo frontale di Agri.q può sterzare seguendo un raggio di curvatura $|R_1| > 0$. Risulta quindi necessario ricavare un'espressione che leghi i raggi di curvatura dei due moduli del robot Agri.q. Si consideri una configurazione generica del robot:

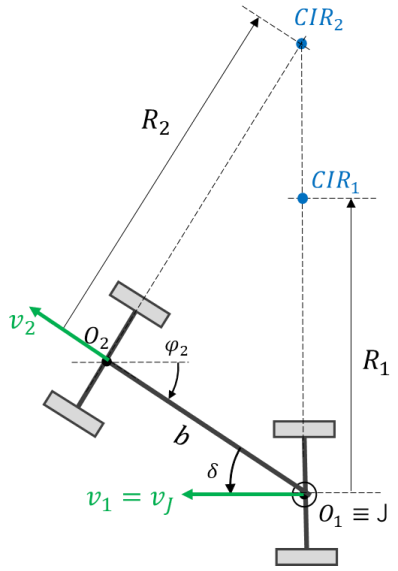

Figura 4.6 – Modello cinematico del robot Agri.q

Nel caso più generale i centri di istantanea rotazione dei due moduli non coincidono in quanto fanno riferimento a due corpi distinti uniti da un giunto. In questo caso, poiché la lunghezza a è nulla, non è possibile trovare una relazione che leghi i due raggi di curvatura basata esclusivamente sulla geometria del robot e sulla configurazione da esso assunta. Infatti, la relazione che lega i raggi di curvatura R_1 ed R_2 può essere determinata solo se si conoscono anche le velocità angolari dei due moduli:

$$R_1 = \sqrt{b^2 + R_2^2} \cdot \frac{\dot{\varphi}_2}{\dot{\varphi}_1}$$

Uno stratagemma per ovviare a questo problema, riuscendo a definire R_1 in funzione unicamente di R_2 e δ, è il seguente:

$$R_1 = \sqrt{b^2 + R_2^2} \cdot \left(1 - \frac{\delta}{\delta_{max}}\right)$$

In questo modo, a parità di R_2, è possibile ridurre il raggio del modulo anteriore, all'aumentare dell'angolo δ, quindi più l'angolo δ si avvicina al valore limite più il modulo anteriore agisce in modo rapido (sterzando con un raggio di curvatura ridotto).
Il raggio R_1 va definito anche in segno. Nel caso più generale il robot potrebbe trovarsi con i due centri di istantanea rotazione sui due lati opposti rispetto al punto O_1 (Figura 4.7), tuttavia, tale configurazione è molto particolare ed in marcia indietro ha la sola funzione di avvicinare il robot al jackknife.

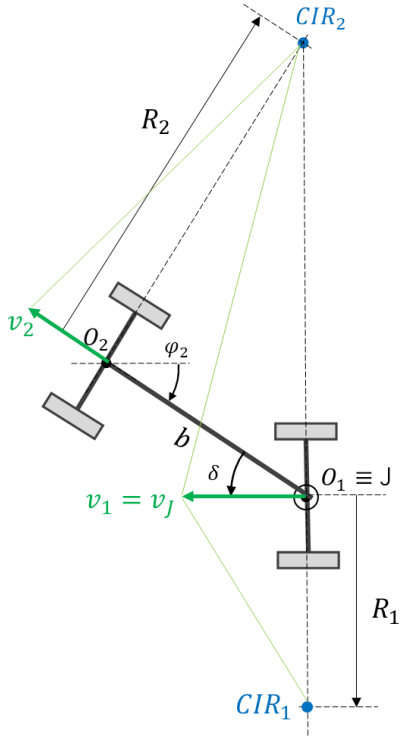

Figura 4.7 - Configurazione generale del robot Agri.q con i CIR situati su lati opposti

Da queste considerazioni è scaturita la scelta di ipotizzare che il segno di R_1 coincida col segno di R_2 (cioè che i moduli ruotino nello stesso verso). Quindi l'espressione che lega i due raggi diventa:

$$R_1 = sign(R_2) \cdot \sqrt{b^2 + R_2^2} \cdot \left(1 - \frac{\delta}{\delta_{max}}\right) \tag{4.8}$$

Per come è strutturato il robot Agri.q non si può verificare una condizione in cui, nonostante si stia ragionando in retromarcia, il modulo frontale si muove in avanti. Infatti, anche nei casi in cui in marcia indietro la componente longitudinale della velocità v_J è diretta in avanti, il moto del modulo anteriore è anch'esso all'indietro.

5. Capitolo 5: Introduzione ai sistemi di controllo del moto in retromarcia

5.1 Controllo del moto in retromarcia per il robot Epi.q

Il sistema di controllo del moto in retromarcia sviluppato per il robot Epi.q può essere riassunto dal seguente schema:

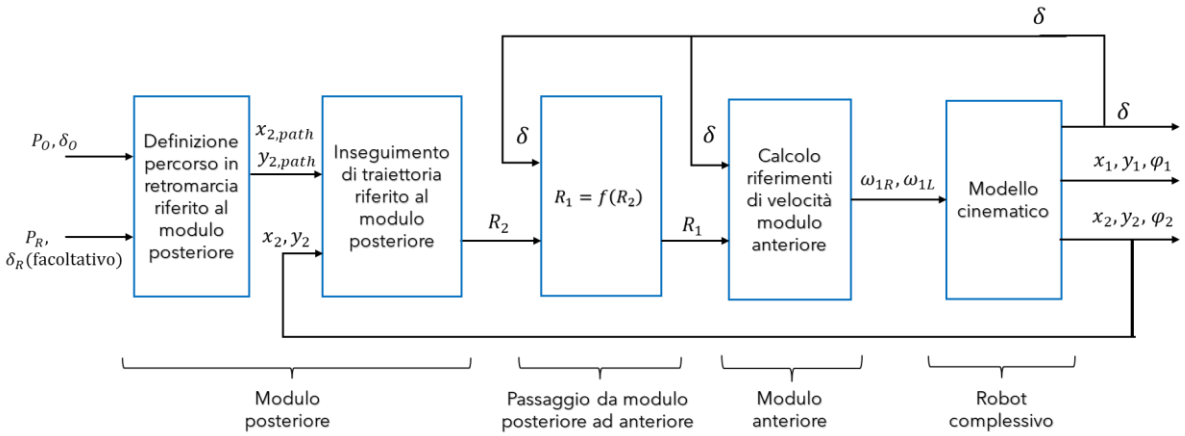

Figura 5.1 – Schema a blocchi del controllore della guida autonoma in retromarcia di Epi.q

Come già indicato nel capitolo precedente, per poter eseguire delle manovre in retromarcia senza perdere il controllo del rimorchio si cambia sistema di riferimento ragionando non più dal punto di vista del modulo fontale (punto O_1) e del raggio di curvatura R_1, ma dal punto di vista del modulo posteriore (punto O_2) e del raggio di curvatura R_2. Partendo da questo presupposto i primi due blocchi del sistema di controllo (pianificazione del percorso ed inseguimento di traiettoria) prendono come riferimento la posa del retrotreno.

La pianificazione del percorso ha lo scopo di determinare un tragitto per portare il robot da una configurazione iniziale $P_O = (x_{2,O},\ y_{2,O},\ \varphi_{2,O})$ ad una configurazione finale $P_R = (x_{2,R},\ y_{2,R},\ \varphi_{2,R})$. Tali configurazioni, nonché tutti i punti che costituiscono il percorso, devono essere compatibili con i vincoli cinematici del robot. In questo caso il vincolo è costituito dal limite δ_{max} sull'angolo di imbardata tra i due moduli. Lo studio della

pianificazione di un percorso ammissibile in retromarcia viene trattato nel Capitolo 6 (percorso in assenza di ostacoli) e nel Capitolo 7 (percorso in presenza di ostacoli).

La pianificazione del percorso in retromarcia fornisce i termini $(x_{2,path}, y_{2,path})$ che il blocco di inseguimento di traiettoria sfrutta come riferimenti al fine di calcolare il raggio di curvatura R_2 che consenta al modulo posteriore di raggiungere e/o mantenere il percorso. Una descrizione dettagliata sul funzionamento dell'inseguitore di traiettoria viene fornita nel Capitolo 8.

Il raggio di curvatura R_2 in uscita dal blocco di inseguimento del percorso viene successivamente convertito nel corrispondente raggio di curvatura R_1 relativo al modulo frontale. Ciò è possibile grazie alle formule (4.6) e (4.8) ricavate nei paragrafi precedenti. Noto R_1 ed il valore dell'angolo δ che il robot ha in quell'istante, il controllo di alto livello ha il compito di determinare i riferimenti di velocità da imporre alle ruote del modulo anteriore (ω_{1R} e ω_{1L}) per ottenere il raggio di curvatura R_1 voluto. Questo compito viene svolto dal blocco '*Calcolo riferimenti di velocità modulo anteriore*' il cui funzionamento viene descritto di seguito. Infine, i rifermenti ω_{1R} e ω_{1L} vengono inviati al modello cinematico del robot in modo da eseguire il percorso in retromarcia richiesto.

5.2 Controllo del moto in retromarcia per il robot Agri.q

Il sistema di controllo sviluppato per Agri.q non si discosta molto da quello definito per Epi.q. I due sistemi infatti differiscono per un aspetto, visibile nello schema sottostante:

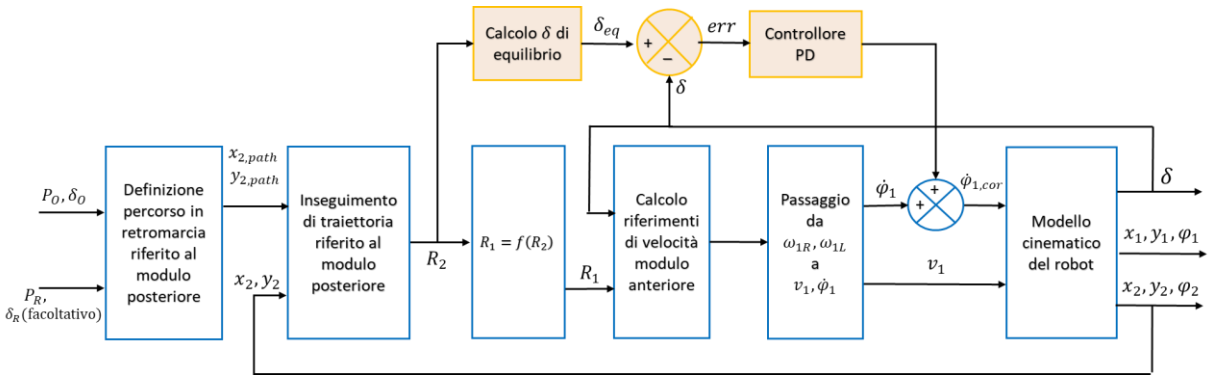

Figura 5.2 – Schema a blocchi del controllore della guida autonoma in retromarcia di Agri.q

Per poter controllare il modulo posteriore di Agri.q nelle fasi di retromarcia è necessario aggiungere il controllo di basso livello, in grado di stabilizzare l'angolo δ attorno al valore δ_{eq}, descritto nella Sezione 3.3.

Come si può vedere dallo schema, il calcolo del δ_{eq} non è funzione ρ_1, come era stato descritto precedentemente, bensì è funzione di R_2. Nota la relazione (4.8) e sapendo che $R_1 = \frac{1}{\rho_1}$ è possibile determinare la nuova espressione per il calcolo del $\delta_{eq} = f(R_2)$.
Il controllo di basso livello, in base al confronto tra δ_{eq} ed il δ in uscita dal modello cinematico, va ad agire sulla velocità angolare del modulo anteriore $\dot{\varphi}_1$ in modo tale da seguire il valore di rifermento δ_{eq}.

In questo modo, anche per il robot Agri.q, è possibile ottenere da un lato l'inseguimento del percorso in retromarcia voluto, dall'altro il controllo dell'angolo δ in modo da evitare il fenomeno del jackknifing.

5.3 Calcolo dei riferimenti di velocità del modulo anteriore

Il blocco *'Calcolo riferimenti di velocità modulo anteriore'* ha lo scopo di convertire il raggio di curvatura R_1 in una coppia di velocità da imporre alle ruote del modulo frontale. Per comprenderne il funzionamento di questo blocco si riprende l'espressione (3.15):

$$R_1 = \frac{i}{2} \cdot \frac{\omega_{1R} + \omega_{1L}}{\omega_{1R} - \omega_{1L}}$$

Questa formula può essere utile se si vuole conoscere la successione dei raggi di curvatura di una traiettoria ottenuta imponendo gli andamenti delle velocità angolari ω_{1R} e ω_{1L} delle ruote. In questo caso però il ragionamento logico è l'inverso: si vogliono trovare le velocità delle singole ruote che consentono al robot di seguire una determinata traiettoria di raggio R_1. Esplicitando questa espressione rispetto al rapporto delle velocità:

$$\frac{\omega_{1R}}{\omega_{1L}} = \frac{R_1 + \frac{i}{2}}{R_1 - \frac{i}{2}} \quad \Rightarrow \begin{cases} \omega_{1R} = \dfrac{R_1 + \frac{i}{2}}{R_1 - \frac{i}{2}} \cdot \omega_{1L} \\ \omega_{1L} = \dfrac{R_1 - \frac{i}{2}}{R_1 + \frac{i}{2}} \cdot \omega_{1R} \end{cases}$$

Per determinare le velocità angolari si impone una certa velocità ω della ruota esterna alla traiettoria (ω_{1R} se la rotazione è verso sinistra, ω_{1L} se la rotazione è verso destra). Fissata la velocità angolare di una delle due ruote e noto il raggio R_1 è possibile risalire alla velocità angolare dell'altra ruota tramite le relazioni scritte sopra.

Se $R_1 > 0$ (rotazione in senso orario in marcia indietro):

- $\omega_{1R} = \omega$
- $\omega_{1L} = \frac{R_1 - \frac{i}{2}}{R_1 + \frac{i}{2}} \cdot \omega_{1R}$

Se $R_1 < 0$ (rotazione in senso antiorario in marcia indietro):

- $\omega_{1L} = \omega$
- $\omega_{1R} = \frac{R_1 + \frac{i}{2}}{R_1 - \frac{i}{2}} \cdot \omega_{1L}$

Se $R_1 = 0$ (rotazione sul posto):

- Se $\delta > 0 \quad \Rightarrow \quad \omega_{1L} = \omega \quad \omega_{1R} = -\omega$
- Se $\delta < 0 \quad \Rightarrow \quad \omega_{1L} = -\omega \quad \omega_{1R} = \omega$

Passare attraverso la velocità ω risulta vantaggioso in quanto si ha sempre il controllo sulla massima velocità angolare (che è sempre quella della ruota più esterna); in questo modo, se le condizioni di moto risultano vicine al limite di velocità (reale o imposto per sicurezza al motore), la velocità longitudinale viene ridotta, ma la velocità angolare voluta è sempre realizzabile. Senza questa accortezza potrebbe verificarsi il caso in cui non è possibile imporre la sterzata voluta perché la velocità longitudinale è troppo alta e non si può accelerare oltre con la ruota esterna.
Inoltre, è possibile impostare l'andamento di ω in modo da ottenere una fase di accelerazione all'inizio del percorso e una fase di decelerazione alla fine. Il limite massimo per il valore di ω è stato posto pari a $0.4\omega_{max}$ poiché una velocità maggiore renderebbe più difficile il controllo dell'angolo δ.

Considerando che si ragiona sempre su un moto in retromarcia, è importante sottolineare che la velocità ω è anche caratterizzata da un segno.
Nel caso del rover Agri.q la velocità ω è sempre negativa in quanto, come è già stato indicato, in marcia indietro il moto del modulo anteriore è anch'esso all'indietro.

Per Epi.q non è detto che ω sia sempre negativa: nel moto in retromarcia il moto del modulo posteriore è all'indietro, mentre la direzione del moto del modulo anteriore non è nota a priori. Riprendendo l'espressione (4.7) è possibile, noti R_1 e δ, definire se il modulo frontale si muove in avanti o all'indietro e quindi stabilire il segno di ω:

- $se\ |\delta| \leq \frac{\pi}{2}$:

  se $-segn(\delta)a\tan\delta < segn(\delta)R_1 \leq 0 \quad \Rightarrow \quad \omega > 0$

  se $segn(\delta)R_1 < -segn(\delta)a\tan\delta \quad o \quad segn(\delta)R_1 \geq 0 \quad \Rightarrow \quad \omega < 0$

- $se\ |\delta| > \frac{\pi}{2}$:

  $se\ segn(\delta)R_1 > -segn(\delta)a\tan\delta \quad o \quad segn(\delta)R_1 \leq 0 \quad \Rightarrow \quad \omega > 0$

  se $-segn(\delta)a\tan\delta < segn(\delta)R_1 \leq 0 \quad \Rightarrow \quad \omega < 0$

Il blocco *'Calcolo riferimenti di velocità modulo anteriore'* ha anche il compito di arrestare il robot nel caso di raggiungimento dell'obiettivo P_{R}.

Dopo aver descritto il modello cinematico dei due robot, i blocchi che legano il raggio di curvatura dei due moduli ed il raggio di curvatura del modulo anteriore alle velocità angolari, nei capitoli successivi si mostrerà il funzionamento dei primi due blocchi, ossia la definizione del percorso e l'inseguimento di quest'ultimo.

6. Capitolo 6: Pianificazione di un percorso in retromarcia

6.1 Strategie di pianificazione

La pianificazione del percorso di un veicolo a guida autonoma consiste nell'individuare una traiettoria che possa portare il robot da una certa configurazione iniziale ad una certa configurazione finale evitando eventuali ostacoli sul percorso e tenendo conto dei vincoli fisici dovuti alla geometria e alla tipologia del veicolo. Per poter determinare completamente la traiettoria, una volta trovato il cammino geometrico ammissibile, occorre scegliere una legge oraria, ovvero il profilo di velocità che deve essere rispettato lungo il percorso.

Esistono diverse strategie di pianificazione. Una prima importante distinzione tra le drive strategy risulta essere la seguente [38][39]:

- elaborazione off-line di un percorso complesso. Il percorso viene elaborato una sola volta all'inizio della missione, a veicolo fermo, in seguito è mantenuto come riferimento da seguire per tutto il tempo, fino alla fine della corsa;
- creazione del percorso in tempo reale e generazione di un comando in grado di realizzarlo. Vengono generati di volta in volta percorsi semplici, basati solo sulla posizione effettiva del veicolo e sulla posizione target.

In questo studio viene trattato il primo caso. Lo svantaggio di questo approccio è che il veicolo non è in grado di gestire situazioni di emergenza o impreviste in quanto non sfrutta l'elaborazione in tempo reale delle traiettorie. Tuttavia, i robot articolati Epi.q ed Agri.q non presentano ancora una sensoristica tale da poter identificare ostacoli o variazioni dell'ambiente circostante in tempo reale.

Questo metodo ha però il vantaggio di poter generare percorsi complessi, poiché non c'è limite di tempo computazionale e tutti i dati disponibili possono essere elaborati con algoritmi sofisticati; infatti, il pianificatore non è parte del controllo ad anello chiuso, ma è solamente il generatore del riferimento utilizzato dall'inseguitore di traiettoria (come visibile nelle Figure 5.1 e 5.2).

Questo approccio è stato anche scelto in quanto si adatta ad ambienti molto strutturati, come i percorsi tra i vigneti nel caso del robot Agri.q o i percorsi in ambienti interni nel caso

di Epi.q; si suppone che la variabilità dinamica di questi ambienti strutturati sia minima se non nulla.

Entrando più nel dettaglio, in questo capitolo viene analizzata la strategia di pianificazione di un percorso che il robot possa eseguire in retromarcia per andare da una configurazione $P_O = (x_{2,O},\ y_{2,O},\ \varphi_{2,O})$ ad una configurazione $P_R = (x_{2,R},\ y_{2,R},\ \varphi_{2,R})$, definite prendendo come riferimento il rimorchio del robot (punto O_2) in virtù delle considerazioni fatte nel Capitolo 3.

Per definire tale percorso è stato scelto, come base di partenza, l'algoritmo di Dubins. Questa logica è stata scelta per la sua semplicità, che si traduce in ridotti tempi di calcolo del percorso, e perché permette di determinare il percorso più breve per collegare tra loro due punti potendo andare solamente in una direzione, non viene quindi presa in considerazione la possibilità di eseguire manovre. Ovviamente l'algoritmo deve essere adattato alle esigenze del caso specifico analizzato in questa tesi, cioè il moto in retromarcia. Quindi si apporteranno delle modifiche al percorso di Dubins in modo da realizzare una pianificazione in marcia indietro.

Un'altra imprescindibile considerazione riguarda il raggio minimo di curvatura: i robot a guida differenziale come Epi.q ed Agri.q sono in grado di effettuare una rotazione sul posto, quindi di sostenere un raggio di curvatura nullo. Tuttavia, poiché l'algoritmo di Dubins è stato pensato considerando veicoli car-like, non è possibile porre $R_{min} = 0\ m$. Per aggirare questo problema è possibile inserire come raggio minimo un valore estremamente piccolo, minore della metà dell'interasse i. Inoltre, come si vedrà successivamente, è raro che in retromarcia vi sia la possibilità di effettuare una rotazione sul posto senza incappare nel fenomeno del jackknife.

6.2 Percorso di Dubins

Si suppongano note la configurazione iniziale del veicolo $P_O = (x_{2,O},\ y_{2,O},\ \varphi_{2,O})$ e la configurazione finale $P_R = (x_{2,R},\ y_{2,R},\ \varphi_{2,R})$, corrispondenti a due punti nel piano euclideo bidimensionale. Si vuole determinare il cammino di lunghezza minima che colleghi questi due punti con un vincolo sulla curvatura del cammino $R_{2,min}$ (pedice 2 in quanto viene preso

come riferimento il modulo posteriore) e con prescritte tangenti iniziali e terminali al cammino. Inoltre, viene aggiunta l'ipotesi che il veicolo possa viaggiare solo in una direzione.

Nel 1957, Lester Eli Dubins [40] ha dimostrato che il percorso più breve per unire questi due punti viene realizzato unendo archi circolari di massima curvatura e linee rette. In particolare, per passare da due configurazioni qualsiasi, il percorso più breve può sempre essere espresso come una combinazione di non più di tre primitive di movimento, le quali possono corrispondere ad una svolta a destra (R), una svolta a sinistra (L) o un tratto rettilineo (S). In altre parole, (R) e (L) rappresentano un tratto di traiettoria corrispondente ad un arco di circonferenza di raggio minimo $R_{2,min}$ percorso in senso orario o antiorario. Le traiettorie ottime vanno quindi cercate tra "parole" costruite con le "lettere" (R), (L), (S). Ogni tratto ha una propria lunghezza:

- t: lunghezza del primo tratto;
- u: lunghezza del secondo tratto;
- v: lunghezza del terzo tratto;

In conclusione, un percorso ottimale per passare da una configurazione iniziale ad una finale può essere espresso come una combinazione di queste tre primitive di movimento:

Tabella 6.1 – Elenco dei 6 diversi tipi di percorso di Dubins

Percorso	**I Tratto (t)**	**II Tratto (u)**	**III Tratto (v)**
1	L	S	L
2	L	S	R
3	R	S	L
4	R	S	R
5	R	L	R
6	L	R	L

Da questa tabella si può osservare come il primo ed il terzo tratto siano sempre dei tratti curvilinei, mentre il tratto centrale può essere un arco di circonferenza o un tratto rettilineo a seconda dei casi. Ad esempio, data una certa posizione iniziale e finale, il percorso ottimale risulta essere del tipo 'RSR'. Questo corrisponde a un arco di svolta a destra (R) seguito da un

segmento di retta (S) seguito da un altro arco di svolta a destra (R). Muovendosi lungo ogni segmento in questa sequenza per la lunghezza appropriata si ottiene la curva più corta che unisce il punto iniziale P_O al punto terminale P_R con le tangenti desiderate e senza superare la curvatura data. Nella Figura 6.1 è riportato un esempio di percorso di Dubins.

Tipo di moto: moto in marcia avanti

Caratteristiche:

$P_O = (0\ m, 0\ m, 0°)$
$P_R = (2\ m, 1\ m, 125°)$
$R_{2,min} = 0.2\ m$
$Forma = 'LSL'$
$t = 0.07\ m$
$u = 1.96\ m$
$v = 0.36\ m$
$L_{tot} = 2.40\ m$
$T_{calcolo} = 0.006\ s$

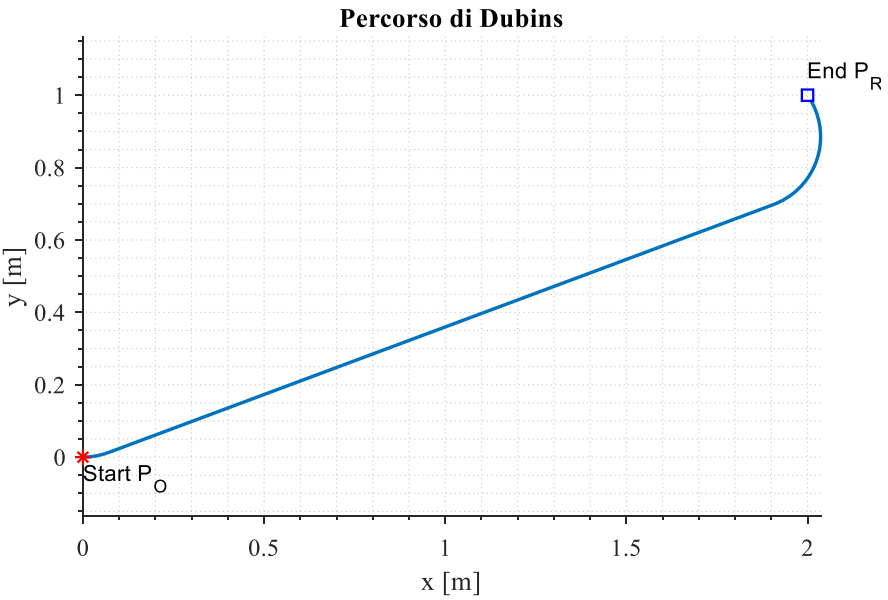

Figura 6.1 – Percorso di Dubins in marcia avanti.
$\boldsymbol{L_{tot}}$: lunghezza totale del percorso, pari alla somma dei tre tratti.
$\boldsymbol{T_{calcolo}}$: tempo necessario per eseguire il calcolo del percorso

È importante notare come l'algoritmo funzioni anche in caso di percorsi semplici come quello mostrato in Figura 6.2. In tal caso il primo ed il terzo tratto risultano essere nulli.

Tipo di moto: moto in marcia avanti

Caratteristiche:

$P_O = (1\ m, 1\ m, 0°)$
$P_R = (2\ m, 1\ m, 0°)$
$R_{2,min} = 0.2\ m$
$t = 0\ m$
$u = 1\ m$
$v = 0\ m$
$L_{tot} = 1\ m$
$T_{calcolo} = 0.012\ s$

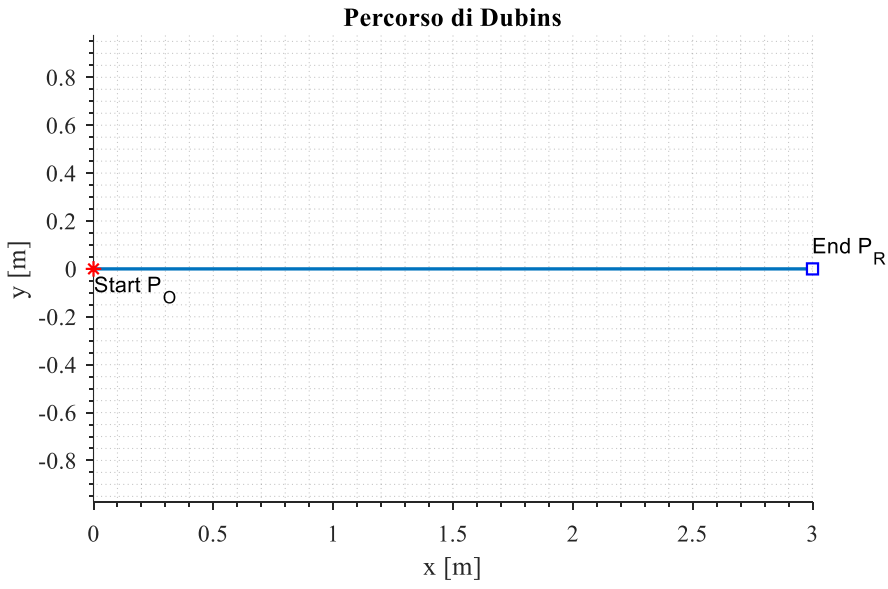

Figura 6.2 – Percorso di Dubins rettilineo in marcia avanti

La semplicità dell'algoritmo di Dubins rende quest'ultimo un ottimo punto di partenza per elaborare logiche di calcolo percorso più complesse e soggette ad un maggior numero di vincoli. Non a caso il percorso di Dubins è comunemente usato nei campi della robotica e della teoria del controllo come un modo per pianificare percorsi per robot su ruote.

6.2.1 Correzione del percorso di Dubins in caso di moto in retromarcia

Il percorso di Dubins ha come ipotesi di partenza il fatto che il veicolo possa raggiungere la configurazione finale muovendosi solo in avanti. Poiché l'obiettivo è invece realizzare un moto all'indietro, risulta necessario effettuare un'ulteriore modifica, rappresentata graficamente in Figura 6.3, in modo tale che il percorso venga pensato per un robot mobile che si muove in retromarcia per passare da un punto iniziale ad un punto finale.

Se l'intento è quello di passare da una configurazione caratterizzata da un angolo $\varphi_{2,O}$ ad una configurazione $\varphi_{2,R}$, l'algoritmo Dubins propone un moto in avanti rappresentato nella Figura 6.3 in blu. Per poter ottenere un modo in retromarcia è sufficiente cambiare sistema di riferimento in modo tale che il moto all'indietro del modulo posteriore corrisponda al moto in avanti visto dal sistema di riferimento Dubins. Ciò si ottiene agendo sulla definizione delle due configurazioni date come input all'algoritmo di Dubins:

$$P_O = \left(x_{2,O},\, y_{2,O},\, \varphi_{2,OD}\right)$$
$$P_R = \left(x_{2,R},\, y_{2,R},\, \varphi_{2,RD}\right)$$

dove

$$\varphi_{2,OD} = \varphi_{2,O} + \pi$$
$$\varphi_{2,RD} = \varphi_{2,R} + \pi$$

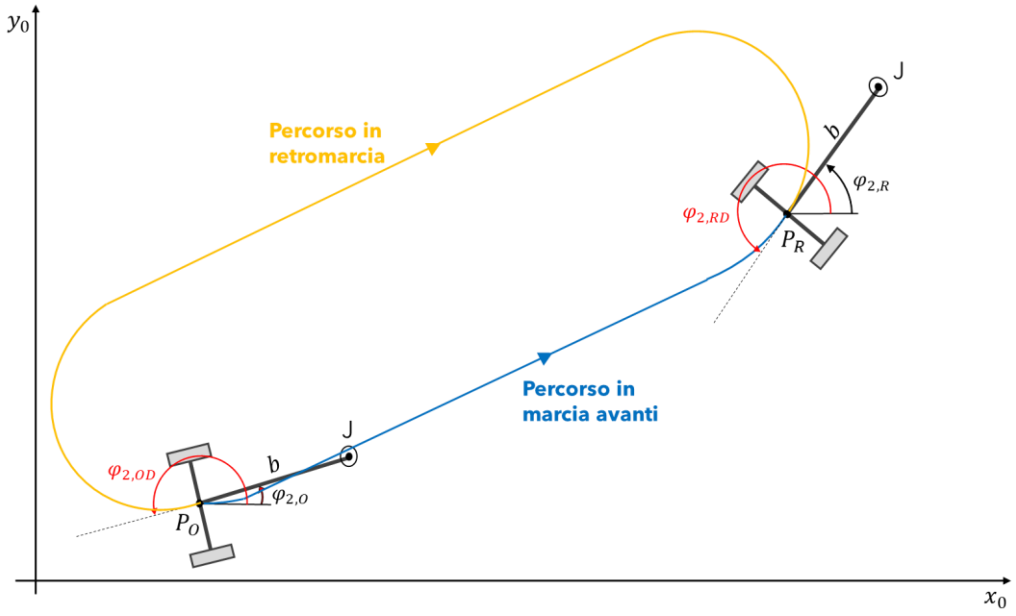

Figura 6.3 - Percorso in marcia avanti o in marcia indietro del modulo posteriore

Viene riportato un esempio:

Input:
$P_O = (2\ m, 1m, 0°)$
$P_R = (5\ m, 3\ m, -135°)$
$R_{2,min} = 0.3\ m$

Caratteristiche moto in marcia avanti:
$Forma$ = 'LSR'
$t = 0.23\ m$
$u = 3.33\ m$
$v = 0.94\ m$
$L_{tot} = 4.50\ m$
$T_{calcolo} = 0.009\ s$

Caratteristiche moto in retromarcia:
$Forma$ = 'RSL'
$t = 0.82\ m$
$u = 3.33\ m$
$v = 0.11\ m$
$L_{tot} = 4.25\ m$
$T_{calcolo} = 0.007\ s$

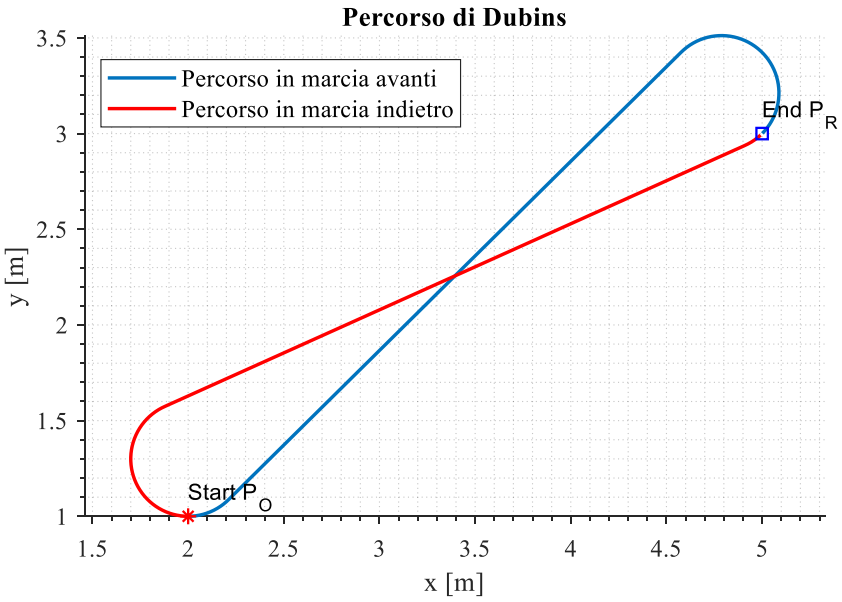

Figura 6.4 – Percorso di Dubins in marcia avanti ed in marcia indietro

6.2.2 Discontinuità di curvatura nei punti di congiunzione tra i tratti

Uno dei difetti di tale percorso è la presenza di discontinuità di curvatura nel passaggio da un tratto all'altro, tale andamento causa a sua volta delle discontinuità delle velocità angolari della ruota destra e sinistra. Se si considera un tratto (S), il robot avanzerà su tale percorso con le due velocità angolari $(\omega_{1R}, \omega_{1L})$ coincidenti in modo da seguire il tratto rettilineo. Invece, se il robot sta percorrendo un tratto curvilineo, la velocità angolare a cui gira la ruota più esterna sarà maggiore della velocità angolare della ruota più interna. Questo vuol dire che nel passaggio tra un tratto rettilineo (S) ed un tratto curvilineo (R) o (L), si ha una discontinuità delle velocità angolari associata alla corrispondente discontinuità di curvatura (in quanto si passa istantaneamente da una curvatura nulla ad una curvatura diversa da 0). Lo stesso vale se si passa istantaneamente da una curva verso destra ad una curva verso sinistra e viceversa.

Avere una velocità che passa istantaneamente da un valore ad un altro equivale ad avere un'accelerazione infinita nel medesimo istante. Poiché i motori hanno un limite sull'accelerazione erogabile, bisogna impostare dei percorsi che non presentino discontinuità di curvatura. A tal proposito su procede ad una correzione del percorso di Dubins mediante la funzione `smoothPathSpline.m`. Queta funzione permette di generare un percorso C^2 continuo (derivate di ordine 0, 1 e 2 continue), interpolando la successione di punti che costituiscono il percorso di Dubins, usando una spline cubica. In questo modo si garantisce la continuità di curvatura.

La curvatura continua consente anche un miglior inseguimento della traiettoria in quanto il controllore dell'angolo δ non è in grado di gestire traiettorie discontinue e quindi in presenza di discontinuità porterebbe ad un'azione correttiva sulla velocità angolare $\dot{\varphi}_1$ più brusca e quindi ad un andamento oscillatorio marcato di $\dot{\varphi}_1$.

Input:
$P_O = (0\ m, 0\ m, 0°)$
$P_R = (3\ m, 1\ m, 120°)$
$R_{2,min} = 1\ m$

NOTA: Non viene rappresentato anche il percorso corretto perché la differenza tra quest'ultimo ed il percorso di Dubins non è apprezzabile.

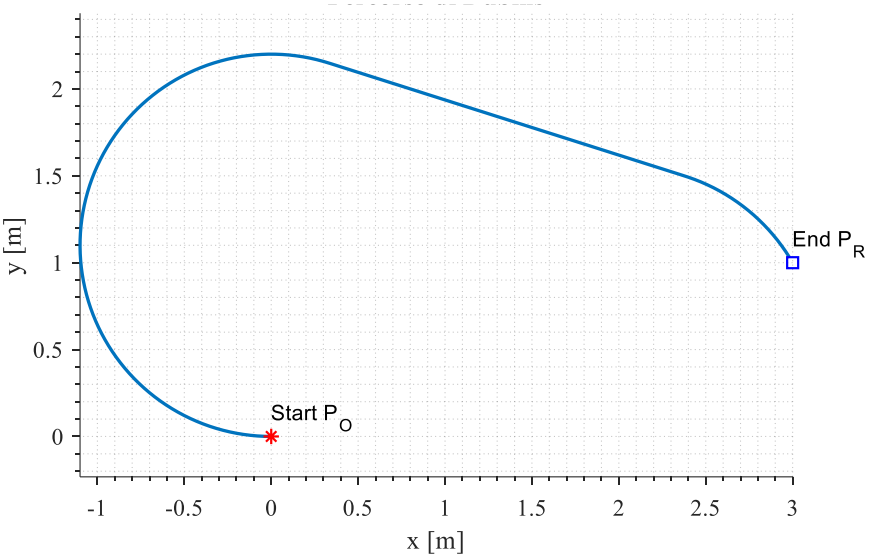

Figura 6.5 – Percorso di Dubins in marcia indietro

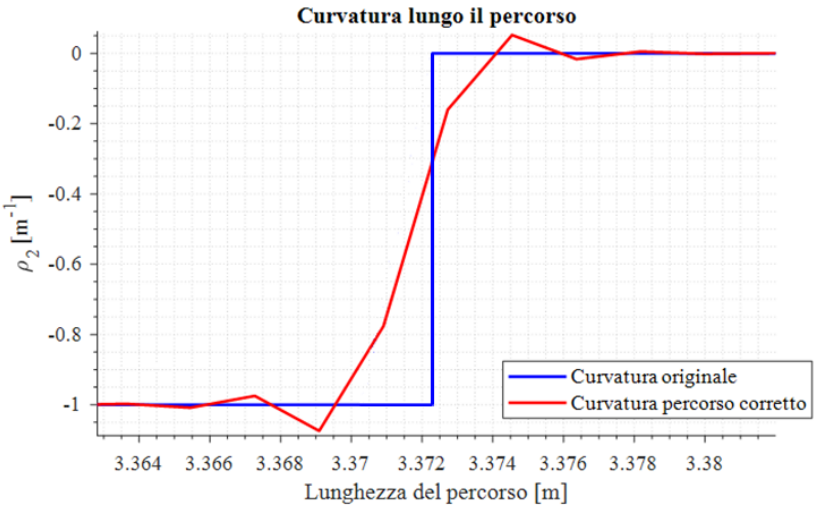

Figura 6.6 – Confronto tra curvatura del percorso di Dubins e curvatura corretta, ingrandimento in corrispondenza di una discontinuità

6.3 Ricerca di un percorso realizzabile dal robot

Quando si progetta una traiettoria bisogna tener conto dei limiti geometrici e cinematici del veicolo oggetto di studio; infatti, può succedere di dover discriminare le traiettorie percorribili da quelle non realizzabili in quanto possono esistere dei limiti in termini di traiettorie possibili.

Questo lavoro vuole sottolineare come il percorso sia direttamente influenzato dai vincoli cinematici imposti dalla geometria e dal tipo di veicolo.

Nei capitoli precedenti è stato descritto come un percorso in retromarcia, curvilineo o rettilineo che sia, possa portare all'urto tra i due moduli che compongono il robot. Una volta determinato il percorso di Dubins, che permette di passare da un punto iniziale ad un punto finale del piano bidimensionale, risulta perciò necessario analizzare tale percorso per poter determinare se sia realizzabile senza incappare nel fenomeno del jackknife.

A questo scopo vengono analizzate delle funzioni, dette funzioni di previsione, che permetto di conoscere in anticipo il comportamento del robot nel percorrere un certo tratto del percorso, in questo caso specifico verranno analizzati i tre tratti proposti dall'algoritmo di Dubins.

Dato il percorso e la configurazione iniziale del robot, le funzioni di previsione permettono di determinare il valore dell'angolo δ assunto lungo esso.

Riprendendo il lavoro svolto nella tesi di Gherlone [19], vengono individuate due funzioni diverse:

1 - $\delta_f = f_1(R, \theta, \delta_i)$ per archi circolari: permette di trovare, noto il valore di δ all'inizio del tratto di percorso, il valore di δ alla fine di questo tratto, rappresentato da una rotazione di un angolo θ e raggio di curvatura costante R.

2 - $\delta_f = f_2(d, \delta_i)$ per tratti rettilinei: permette di individuare, noto il valore dell'angolo δ iniziale, il valore finale di δ dopo un tratto rettilineo di lunghezza d.

L'uso combinato delle due funzioni permette di prevedere l'angolo δ del rimorchio nei punti salienti di una traiettoria composta da archi e segmenti. Infatti, è possibile affermare che, per un tratto rettilineo, in caso di moto in retromarcia il valore di δ maggiore sarà sempre quello alla fine del tratto (se non si considera l'azione di nessun controllo sull'angolo δ). Se invece il robot percorre un arco circolare in retromarcia, l'angolo δ maggiore si forma in uno dei due estremi della traiettoria (sempre senza considerare l'azione di un controllore). Per questo è sufficiente prevedere il valore di δ alla fine di ogni tratto Dubins, dato che la configurazione iniziale originaria è verificata.

6.3.1 Funzione di previsione f1

Prima di considerare il moto in retromarcia, viene ripreso lo studio [19], svolto considerando un veicolo articolato che si muove in marcia avanti lungo un tratto curvilineo:

dato un tratto di curva di raggio costante R_1 che sottende un angolo θ e conoscendo l'angolo iniziale δ_i, la funzione calcola il valore dell'angolo δ_f a fine curva.

Per semplificare i calcoli per l'ottenimento della funzione, si effettua prima lo studio assumendo $R_1 = 0$, dopodiché si estendono i risultati per R_1 qualsiasi. In altre parole, si intende ora calcolare la funzione $\delta_f = f_1(\theta, \delta_i)$ che permetta di conoscere il valore di δ dopo aver compiuto una rotazione sul posto.

Viene analizzata la situazione in cui il veicolo, da una configurazione iniziale generica, attua una rotazione infinitesima di raggio nullo attorno al punto O_1 (Figura 6.7).

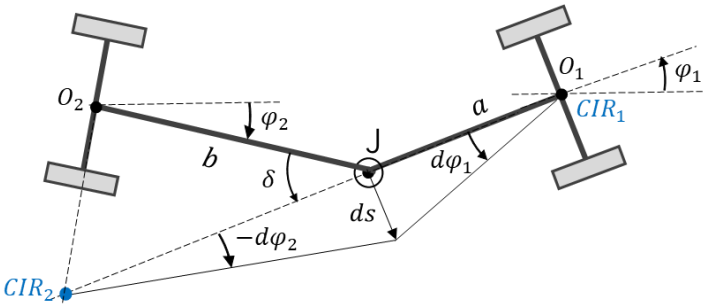

Figura 6.7 – Rotazione sul posto infinitesima $d\varphi_1$ del modulo anteriore

Alla rotazione infinitesima $d\varphi_1$ del modulo anteriore attorno a O_1 (dove $O_1 = CIR_1$ in questo caso) corrisponde una rotazione infinitesima $-d\varphi_2$ del rimorchio attorno al proprio centro di istantanea rotazione CIR_2. L'incremento è negativo poiché il rimorchio ruota in senso orario, al contrario del modulo frontale. L'incremento risultante $d\delta$ risulta essere la somma dei due incrementi di modulo anteriore e posteriore:

$$d\delta = d\varphi_1 - d\varphi_2$$

Esprimendo lo spostamento infinitesimo ds del giunto sia secondo grandezze di un modulo che dell'altro, si ottiene un'equazione unica che mette in relazione tra loro le grandezze in gioco.

$$\begin{cases} ds = a\ d\varphi_1 \\ ds = \dfrac{b}{\cos\delta}\ (-d\varphi_2) \end{cases} \quad \Rightarrow \quad a\ d\varphi_1 = \frac{b}{\cos\delta}\ (-d\varphi_2)$$

Sostituendo a $-d\varphi_2$ l'espressione contenente $d\delta$:

$$a\ d\varphi_1 = \frac{b}{\cos\delta}\ (d\delta - d\varphi_1)$$

Esplicitando l'equazione che si ottiene rispetto a $d\varphi_1$ ed integrandola lungo un'intera rotazione θ, da $\varphi_{1,i}$ a $\varphi_{1,f}$, con δ che passa da δ_i a δ_f:

$$\theta = \varphi_{1,f} - \varphi_{1,i} = b \int_{\delta_i}^{\delta_f} \frac{1}{b + a\cos\delta} d\delta$$

La soluzione dell'integrale a secondo membro varia a seconda del valore di a e b. Poiché sia per Agri.q che per Epi.q $a < b$ la soluzione da considerare è la seguente:

$$\theta = \frac{2b}{\sqrt{b^2 - a^2}} \left[arctan \frac{(b-a)\, tan\frac{\delta_f}{2}}{\sqrt{b^2 - a^2}} - arctan \frac{(b-a)\, tan\frac{\delta_i}{2}}{\sqrt{b^2 - a^2}} \right]$$

La funzione trovata è del tipo $\theta = f(\delta_f, \delta_i)$. La funzione che si sta cercando però è $\delta_f = f_1(\theta, \delta_i)$, per cui bisogna esplicitare le soluzioni trovate rispetto a δ_f. Per il caso $a < b$ l'inversione della funzione porta alla seguente soluzione:

$$\delta_f = 2\, arctan \left[\frac{\sqrt{b^2 - a^2}}{b - a} tan \left(\frac{\sqrt{b^2 - a^2}}{2b} \theta + arctan \frac{(b-a)\, tan\frac{\delta_i}{2}}{\sqrt{b^2 - a^2}} \right) \right] \qquad (6.1)$$

Si può così prevedere il comportamento del rimorchio lungo una rotazione sul posto attorno al punto O_1.

Si tratta ora di estendere la validità della funzione a curve di raggio R_1 non nullo. Per farlo, consideriamo lo schema seguente:

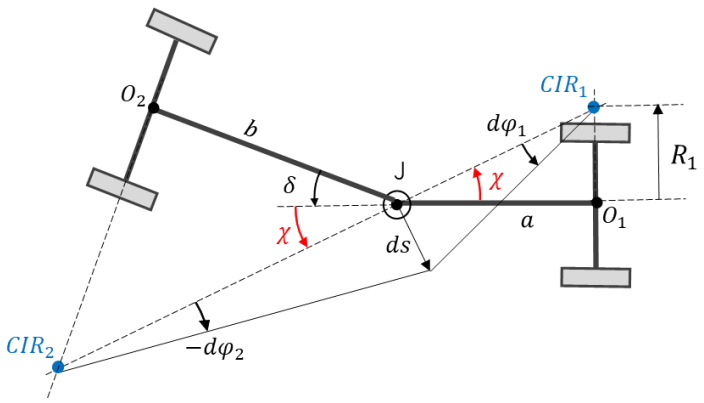

Figura 6.8 – Rotazione infinitesima $d\varphi_1$ di raggio R_1 attorno a CIR_1

Il veicolo è raffigurato in una generica configurazione, nell'istante in cui compie una traiettoria di raggio di curvatura R_1 costante. Anche in questo caso si scrivono le espressioni dello spostamento infinitesimo ds e si uniscono tra loro:

$$\begin{cases} ds = \dfrac{a}{\cos\chi}\ d\varphi_1 \\ ds = \dfrac{b}{\cos(\delta+\chi)}\ (-d\varphi_2) \end{cases} \quad \Rightarrow \quad \frac{a}{\cos\chi}\ d\varphi_1 = \frac{b}{\cos(\delta+\chi)}\ (-d\varphi_2)$$

Sostituendo $-d\varphi_2$ si ottiene:

$$\frac{a}{\cos\chi}\ d\varphi_1 = \frac{b}{\cos(\delta+\chi)}\ (d\delta - d\varphi_1)$$

Poiché l'angolo χ è direttamente correlato a R_1 e quindi è costante ($R_1 = a\tan\chi$), si può procedere ad un confronto fra questa equazione e quella analoga ricavata per la semplice rotazione sul posto, riproposta qui di seguito:

$$a\ d\varphi_1 = \frac{b}{\cos\delta}\ (d\delta - d\varphi_1)$$

In quest'ultima basta fare due sostituzioni per ottenere l'equazione precedente:

$$\begin{cases} a & \Rightarrow & \dfrac{a}{\cos\chi} \\ \delta & \Rightarrow & \delta+\chi \end{cases}$$

La prima sostituzione è evidente, la seconda meno in quanto nell'equazione finale compare oltre a δ anche $d\delta$. Per dimostrare quest'ultima basta notare che l'espressione $d\delta = d(\delta+\chi)$ è verificata, in quanto il differenziale di una costante (χ) è nullo.

Appurata la validità di tali sostituzioni, invece di integrare l'equazione ottenuta come fatto precedentemente, si possono sfruttare direttamente le soluzioni trovate e operare direttamente in esse le sostituzioni. La condizioni che racchiudono i tre casi di soluzione risultano perciò:

$$\frac{a}{\cos\chi} > b, \quad \frac{a}{\cos\chi} < b, \quad \frac{a}{\cos\chi} = b$$

I robot Epi.q ed Agri.q rientrano nel secondo caso, per cui la soluzione è:

$$\delta_f = -\chi + 2\,arctan\left[\frac{\sqrt{b^2 - \left(\frac{a}{cos\,\chi}\right)^2}}{b - \frac{a}{cos\,\chi}} tan\left(\frac{\sqrt{b^2 - \left(\frac{a}{cos\,\chi}\right)^2}}{2b}\theta + arctan\frac{\left(b - \frac{a}{cos\,\chi}\right) tan\frac{\delta_i + \chi}{2}}{\sqrt{b^2 - \left(\frac{a}{cos\,\chi}\right)^2}}\right)\right] \quad (6.2)$$

Le soluzioni degli altri casi sono disponibili in [19].

La funzione $\delta_f = f_1(R_1, \theta, \delta_i)$ ottenuta è uno strumento potente in quanto è adattabile a qualunque traiettoria di cui si conosca il raggio di curvatura costante R_1 e l'angolo sotteso θ. La funzione risulta valida anche per il caso di rotazione sul posto.

6.3.2 Funzione di previsione f1 in caso di moto in retromarcia

Per poter controllare il robot in caso di moto in retromarcia si è visto che un approccio possibile è quello di prendere come riferimento non più il modulo anteriore ed il raggio di curvatura R_1, ma di concentrare l'attenzione sul modulo posteriore (punto O_2) e sul raggio di curvatura R_2. Quindi se il robot si muove all'indietro occorre passare dalla funzione di previsione $\delta_f = f_1(R_1, \theta, \delta_i)$ alla funzione $\delta_f = f_1(R_2, \theta, \delta_i)$ che prenda come grandezze di input il raggio di curvatura e l'arco di circonferenza che caratterizzano il percorso seguito dal modulo posteriore per determinare il valore di δ_f.

Come per il caso in avanti, si analizza prima il caso di rotazione sul posto del modulo posteriore ($R_2 = 0$).

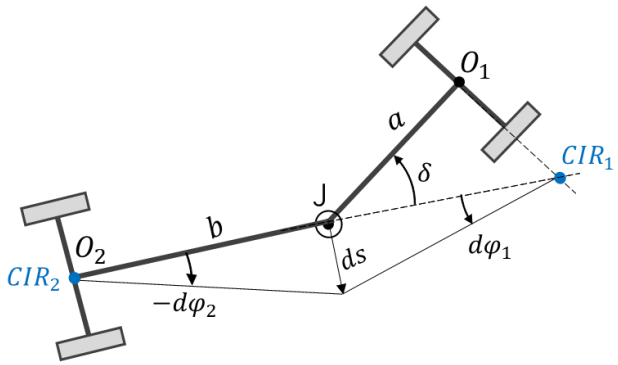

Figura 6.9 – Rotazione sul posto infinitesima $d\varphi_2$ del modulo posteriore

Alla rotazione infinitesima $-d\varphi_2$ del modulo posteriore attorno a O_2 (= CIR_2) corrisponde la rotazione infinitesima $d\varphi_1$ del modulo anteriore attorno al proprio centro di istantanea

rotazione CIR_1. L'incremento $d\varphi_2$ è negativo perché il modulo posteriore ruota in senso orario, l'incremento $d\varphi_1$ è positivo perché il modulo anteriore ruota in senso antiorario.

Anche in retromarcia vale sempre la relazione:

$$d\delta = d\varphi_1 - d\varphi_2$$

Studiando la Figura 6.9 si può esprimere lo spostamento ds sia dal punto di vista del rimorchio che dal punto di vista del modulo anteriore:

$$\begin{cases} ds = b\,(-d\varphi_2) \\ ds = \dfrac{a}{\cos\delta} d\varphi_1 \end{cases} \quad \Rightarrow \quad b\,(-d\varphi_2) = \frac{a}{\cos\delta} d\varphi_1$$

Sostituendo a $d\varphi_1$ l'espressione contenente $d\delta = d\varphi_1 - d\varphi_2$, si ottiene:

$$b\,(-d\varphi_2) = \frac{a}{\cos\delta}(d\delta + d\varphi_2) \tag{6.3}$$

Esplicitando rispetto a $d\varphi_2$:

$$d\varphi_2 = -\frac{a}{a + b\cos\delta} d\delta$$

Integriamo l'equazione lungo un'intera rotazione θ, da $\varphi_{2,i}$ a $\varphi_{2,f}$, con δ che passa da δ_i a δ_f.

$$\theta = \varphi_{2,f} - \varphi_{2,i} = -\int_{\delta_i}^{\delta_f} \frac{a}{a + b\cos\delta} d\delta$$

Si procede con un confronto tra l'espressione appena trovata e l'integrale equivalente ottenuto nel caso in marcia avanti:

$$\theta = \int_{\delta_i}^{\delta_f} \frac{b}{b + a\cos\delta} d\delta \qquad \leftrightarrow \qquad \theta = -\int_{\delta_i}^{\delta_f} \frac{a}{a + b\cos\delta} d\delta$$

Trascurando in prima analisi il segno negativo del secondo integrale, si nota che le due espressioni presentano la stessa struttura, ma con i termini a e b invertiti. Ciò significa che è possibile riprendere la soluzione dell'integrale ottenuta per il moto in avanti e adattarla, adottando le opportune correzioni, al caso in retromarcia.

I robot Epi.q ed Agri,q sono caratterizzati dall'avere la lunghezza $a < b$. Poiché, passando dallo studio del moto in avanti allo studio del moto all'indietro, i termini a e b si sono

invertiti, il caso $a < b$ per il moto in marcia indietro corrisponde al caso $a > b$ per il moto in marcia avanti, con l'aggiunta di un segno "-".

Per cui la soluzione dell'integrale per $a < b$ diventa:

$$\theta = -\frac{a}{\sqrt{b^2-a^2}}\left[\ln\left|\frac{(b-a)\tan\frac{\delta_f}{2}+\sqrt{b^2-a^2}}{(b-a)\tan\frac{\delta_f}{2}-\sqrt{b^2-a^2}}\right| - \ln\left|\frac{(b-a)\tan\frac{\delta_i}{2}+\sqrt{b^2-a^2}}{(b-a)\tan\frac{\delta_i}{2}-\sqrt{b^2-a^2}}\right|\right]$$

Nota la funzione $\theta = f(\delta_f, \delta_i)$, si procede con la ricerca della funzione $\delta_f = f_1(\theta, \delta_i)$. In questo caso però la variabile δ_f è presente nell'argomento di un modulo, rendendo il processo di inversione matematicamente impossibile, dato che l'incognita è funzione di sé stessa. Analizzando comunque il segno dell'argomento del modulo, si ricava che:

- se $-\delta_u < \delta_f < \delta_u \quad \Rightarrow$ argomento modulo positivo
- se $\delta_f > \delta_u \ o \ \delta_f < -\delta_u \quad \Rightarrow$ argomento modulo negativo

dove δ_u è il valore ultimo che δ assume con una rotazione sul posto infinita, in formula:

$$a = b\cos(\pi - \delta_u)$$

$$\delta_u = \cos^{-1}\left(-\frac{a}{b}\right)$$

valore sempre esistente in quanto $b > a$.

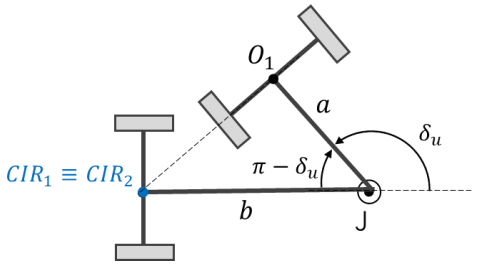

Figura 6.10 – Configurazione dopo rotazione sul posto infinita

Il valore δ_u funge in pratica da asintoto per gli andamenti di δ durante la rotazione sul posto (Figura 6.10); se l'angolo iniziale δ_i è minore di δ_u in valore assoluto, all'infinito δ tenderà ad avvicinarsi a δ_u provenendo da valori inferiori (δ_u^-), mentre se δ_u è maggiore di δ_u in valore assoluto, all'infinito δ si avvicinerà a δ_u provenendo da valori superiori (δ_u^+). Ciò

significa che durante la rotazione l'angolo δ non attraversa mai il valore δ_u, per cui se $\delta_i < \delta_u$ anche $\delta_f < \delta_u$, e lo stesso per il segno maggiore.

Detto ciò, le condizioni che decidono il segno dell'argomento del modulo si possono trasformare:

$$\begin{cases} -\delta_u < \delta_f < \delta_u \quad \Rightarrow \quad -\delta_u < \delta_i < \delta_u \\ \delta_f > \delta_u \ o \ \ \delta_f < -\delta_u \quad \Rightarrow \quad \delta_i > \delta_u \ o \ \ \delta_i < -\delta \end{cases}$$

La sostituzione di δ_f con δ_i non comporta nessuna perdita di informazioni e permette di invertire la funzione anche in presenza dell'incognita nel modulo, aggirando così l'ostacolo matematico del problema.

Esplicitando, per il caso $a < b$ le funzioni $\delta_f = f_1(\theta, \delta_i)$ dei due sottocasi diventano raggruppabili nella seguente:

$$\delta_f = 2arctan\left[\frac{\sqrt{b^2-a^2}}{b-a} \cdot \frac{-1 \pm e^{-\theta\frac{1}{b}\sqrt{b^2-a^2}+\ln\left|\frac{(b-a)\tan\frac{\delta_i}{2}+\sqrt{b^2-a^2}}{(b-a)\tan\frac{\delta_i}{2}-\sqrt{b^2-a^2}}\right|}}{1 \pm e^{-\theta\frac{1}{b}\sqrt{b^2-a^2}+\ln\left|\frac{(b-a)\tan\frac{\delta_i}{2}+\sqrt{b^2-a^2}}{(b-a)\tan\frac{\delta_i}{2}-\sqrt{b^2-a^2}}\right|}}\right] \tag{6.4}$$

$$\begin{cases} se \quad [-\delta_u < \delta_i < \delta_u] \quad \Rightarrow \quad segno \ + \\ se \quad [\delta_i > \delta_u \ o \ \ \delta_i < -\delta_u] \quad \Rightarrow \quad segno \ - \end{cases}$$

Tramite la funzione appena trovata è possibile prevedere il comportamento del modulo posteriore in caso di rotazione sul posto attorno al punto O_2.

Si procede con all'analisi del comportamento del modulo posteriore lungo una traiettoria di raggio $R_2 \neq 0$ e costante. Si considera la Figura seguente:

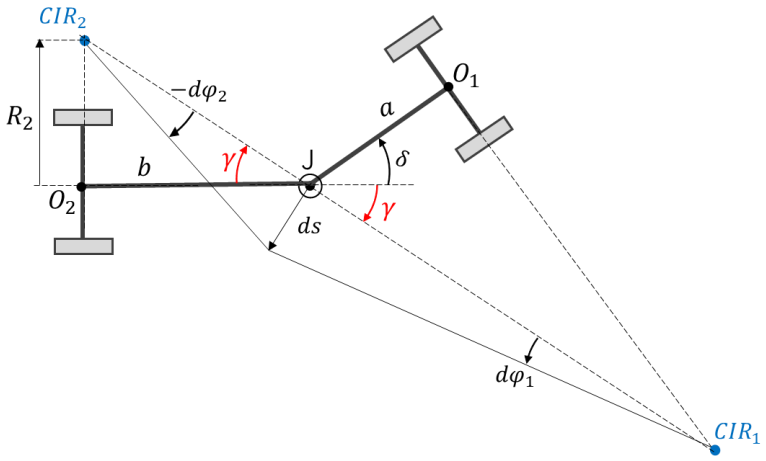

Figura 6.11 – Rotazione infinitesima $d\varphi_2$ di raggio R_2 attorno a CIR_2

Scrivendo nuovamente le espressioni dello spostamento infinitesimo ds:

$$\begin{cases} ds = \dfrac{b}{\cos\gamma}\,(-d\varphi_2) \\ ds = \dfrac{a}{\cos(\delta+\gamma)}\,d\varphi_1 \end{cases} \quad \Rightarrow \quad \frac{b}{\cos\gamma}\,(-d\varphi_2) = \frac{a}{\cos(\delta+\gamma)}\,d\varphi_1$$

Sostituendo $d\varphi_1$:

$$\frac{b}{\cos\gamma}\,(-d\varphi_2) = \frac{a}{\cos(\delta+\gamma)}\,(d\delta + d\varphi_2)$$

Anche in questo caso l'angolo γ è strettamente legato al raggio R_2 in quanto:

$$R_2 = b\tan\gamma \quad \rightarrow \quad \gamma = \tan^{-1}\left(\frac{R_2}{b}\right)$$

Quindi se il raggio di curvatura R_2 è costante anche l'angolo γ è costante. Si può quindi procedere al confronto fra l'equazione (6.3) e la corrispondente equazione ricavata per la rotazione sul posto:

$$b\,(-d\varphi_2) = \frac{a}{\cos\delta}\,(d\delta + d\varphi_2) \quad \leftrightarrow \quad \frac{b}{\cos\gamma}\,(-d\varphi_2) = \frac{a}{\cos(\delta+\gamma)}\,(d\delta + d\varphi_2)$$

Anche in questo caso basta fare due sostituzioni per passare da un'equazione all'altra:

$$\begin{cases} b & \rightarrow & \dfrac{b}{\cos\gamma} \\ \delta & \rightarrow & \delta+\gamma \end{cases}$$

Il caso $\frac{b}{\cos\gamma} > a$ è quello in cui ricadono i robot Epi.q ed Agri.q.

Anche in caso di marcia indietro si possono sfruttare direttamente le soluzioni trovate nel caso di rotazione sul posto e operare direttamente su esse le sostituzioni. Nel caso di rotazione sul posto erano presenti due sottocasi dovuti la presenza del modulo, si ha un ulteriore casistica a seconda del segno del raggio R_2 (uguale al segno di γ):

$$\begin{cases} R \geq 0 & \begin{cases} (1) & -\delta_u - 2\gamma < \delta_i < \delta_u \\ (2) & \delta_i < -\delta_u - 2\gamma \quad o \quad \delta_i > \delta_u \end{cases} \\ R < 0 & \begin{cases} (3) & -\delta_u < \delta_i < \delta_u - 2\gamma \\ (4) & \delta_i < -\delta_u \quad o \quad \delta_i > \delta_u - 2\gamma \end{cases} \end{cases}$$

Il valore δ_u assume ora il significato di angolo a cui tende δ dopo una rotazione infinita di raggio costante R_2.

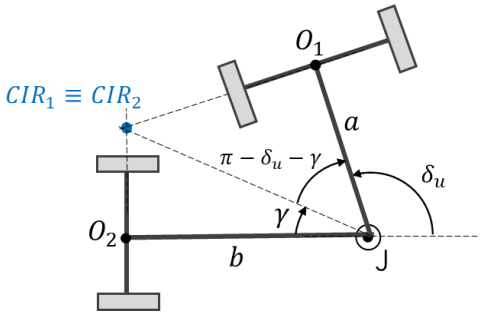

Figura 6.12 – Configurazione dopo rotazione infinita di raggio costante R_2

$$\frac{b}{\cos\gamma} = \frac{a}{\cos(\pi - \delta_u - \gamma)}$$

$$\cos(\delta_u + \gamma) = -\frac{a}{b}\cos\gamma$$

$$\delta_u = -\gamma + \cos^{-1}\left(-\frac{a}{b}\cos\gamma\right)$$

La funzione $\delta_f = f_1(R_2, \theta, \delta_i)$ per il caso $\frac{b}{\cos\gamma} > a$ risulta perciò la seguente:

$$\delta_f = -\gamma + 2\arctan\left[\frac{\sqrt{\left(\frac{b}{\cos\gamma}\right)^2 - a^2}}{\frac{b}{\cos\gamma} - a} \cdot \frac{-1 \pm e^{-\frac{1}{b}\theta\sqrt{\left(\frac{b}{\cos\gamma}\right)^2 - a^2}} \ln\left|\frac{\left(\frac{b}{\cos\gamma} - a\right)\tan\frac{\delta_i+\gamma}{2} + \sqrt{\left(\frac{b}{\cos\gamma}\right)^2 - a^2}}{\left(\frac{b}{\cos\gamma} - a\right)\tan\frac{\delta_i+\gamma}{2} - \sqrt{\left(\frac{b}{\cos\gamma}\right)^2 - a^2}}\right|}{1 \pm e^{-\frac{1}{b}\theta\sqrt{\left(\frac{b}{\cos\gamma}\right)^2 - a^2}} \ln\left|\frac{\left(\frac{b}{\cos\gamma} - a\right)\tan\frac{\delta_i+\gamma}{2} + \sqrt{\left(\frac{b}{\cos\gamma}\right)^2 - a^2}}{\left(\frac{b}{\cos\gamma} - a\right)\tan\frac{\delta_i+\gamma}{2} - \sqrt{\left(\frac{b}{\cos\gamma}\right)^2 - a^2}}\right|}\right] \quad (6.5)$$

Nei sottocasi (1) e (3) si utilizza il segno "+", mentre nei sottocasi (2) e (4) il segno "-".

6.3.3 Funzione di previsione f2

La seconda funzione di previsione si occupa di determinare l'angolo δ che si ottiene al termine di uno spostamento rettilineo di lunghezza d, noto l'angolo iniziale δ_i. Anche in questo caso viene ripreso lo studio di Gherlone [19] svolto per il caso di moto in marcia avanti. Successivamente verrà analizzato il caso di moto rettilineo in retromarcia.

Si considera perciò il robot mobile mentre si muove in marcia avanti.

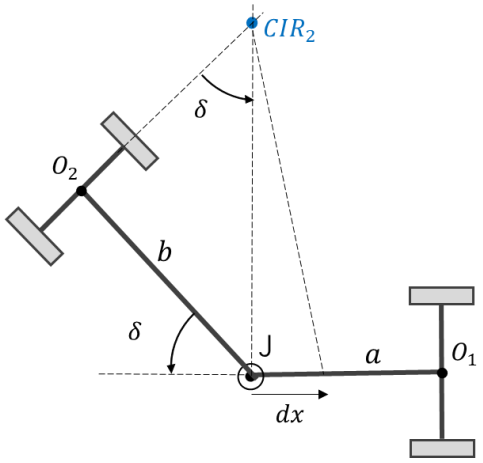

Figura 6.13 – Spostamento del modulo anteriore lungo un tratto rettilineo in marcia avanti

Osservando il modello in Figura 6.13, lo spostamento infinitesimi dx è esprimibile come:

$$dx = -\frac{b}{\sin\delta}d\delta$$

Integrando questa relazione:

$$\int_0^d dx = \int_{\delta_i}^{\delta_f} -\frac{b}{\sin\delta}d\delta \qquad d = b\int_{\delta_f}^{\delta_i}\frac{1}{\sin\delta}d\delta$$

Risolvendo l'integrale a secondo membro si ottiene:

$$d = -\frac{b}{2}\ln\left(\frac{1+\cos\delta_i}{1-\cos\delta_i}\cdot\frac{1-\cos\delta_f}{1+\cos\delta_f}\right)$$

Esplicitando la formula di δ_f rispetto a δ_i si ottiene la funzione di previsione voluta:

$$\delta_f = segn(\delta_\mathrm{i})\cdot\cos^{-1}\left(\frac{1-\frac{1-\cos\delta_\mathrm{i}}{1+\cos\delta_\mathrm{i}}e^{-\frac{2d}{b}}}{1+\frac{1-\cos\delta_\mathrm{i}}{1+\cos\delta_\mathrm{i}}e^{-\frac{2d}{b}}}\right) \tag{6.6}$$

Il termine $segn(\delta_\mathrm{i})$ deve essere aggiunto in quanto la funzione arcocoseno vale solo nell'intervallo $[0,\pi]$. È possibile utilizzare questo espediente in quanto il segno di δ_f corrisponde sempre al segno di δ_i.

6.3.4 Funzione di previsione f2 in caso di moto in retromarcia

La funzione di previsione $\delta_f = f_2(d, \delta_i)$ va modificata se si considera un moto in retromarcia. Infatti, la funzione (6.6), così come viene scritta, porta ad avere sempre una riduzione dell'angolo δ. Come mostrato dalle simulazioni presenti nel Capitolo 3, ciò è vero nel caso di moto in avanti, ma non per il moto all'indietro, in cui si ha un aumento progressivo di δ che, se non controllato, porta all'urto tra i due moduli del robot.

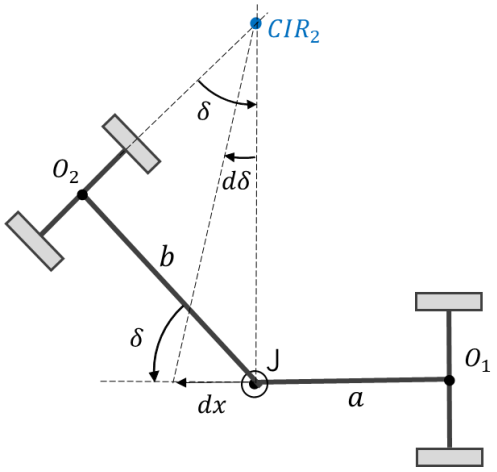

Figura 6.14 – Spostamento del modulo anteriore lungo un tratto rettilineo in marcia indietro

Se si considera il robot mentre si sta muovendo in marcia indietro si può vedere che:

$$dx = \frac{b}{\sin\delta} d\delta$$

Quindi in caso di moto all'indietro si ottiene a secondo membro un termine positivo (a differenza del caso precedente in cui era negativo). Questo in virtù del fatto che in retromarcia l'angolo δ aumenta all'aumentare dello spostamento dx.

Integrando questa espressione

$$\int_0^d dx = \int_{\delta_i}^{\delta_f} \frac{b}{\sin\delta} d\delta$$

E risolvendo l'integrale si ottiene:

$$d = \frac{b}{2} \ln \left| \frac{1+\cos\delta_i}{1-\cos\delta_i} \cdot \frac{1-\cos\delta_f}{1+\cos\delta_f} \right|$$

Esplicitando δ_f rispetto a δ_i si ricava la funzione cercata $\delta_f = (d, \delta_i)$:

$$\delta_f = segn(\delta_\mathrm{i}) \cdot \cos^{-1}\left(\frac{1 - \frac{1-\cos\delta_\mathrm{i}}{1+\cos\delta_\mathrm{i}} e^{\frac{2d}{b}}}{1 + \frac{1-\cos\delta_\mathrm{i}}{1+\cos\delta_\mathrm{i}} e^{\frac{2d}{b}}}\right) \tag{6.7}$$

Anche in questo caso deve essere aggiunto alla funzione il fattore $segn(\delta_\mathrm{i})$.

È importante notare che le funzioni di previsione trovate valgono non solo per i robot Epi.q ed Agri.q, ma anche per tutti quei robot mobili o veicoli che presentano una struttura simile (con $a < b$).

6.4 Funzionamento dell'algoritmo Optimal Path

In questo paragrafo viene analizzato il funzionamento dell'algoritmo Optimal Path.

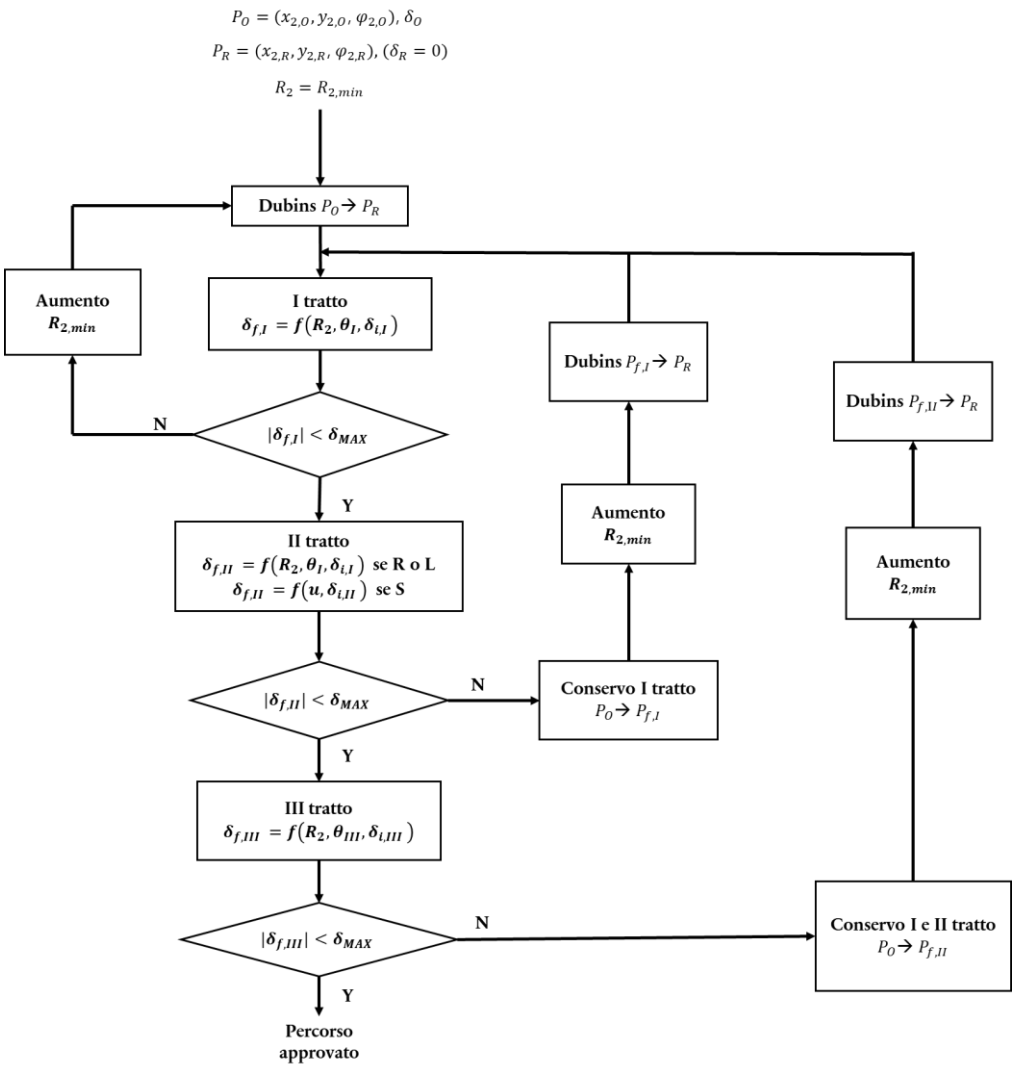

Figura 6.15 – Schema rappresentante il funzionamento dell'algoritmo Optimal Path

Come si può vedere dallo schema in Figura 6.15, l'algoritmo Optimal Path ha lo scopo di analizzare i tre tratti del percorso Dubins in marcia indietro e, mediante le funzioni di previsione descritte nei paragrafi precedenti, determinare se ogni tratto porta al verificarsi del jackknifing o meno. Nel caso in cui il tratto oggetto di analisi porti all'urto tra i due moduli del robot, allora l'algoritmo ha anche il compito di correggere tale tratto.

In questa fase vengono analizzati i tratti prima di applicate le correzioni per rimuovere le discontinuità di curvatura. Le modifiche che vengono successivamente applicate per ottenere la continuità di curvatura non incidono sulle caratteristiche del percorso ottenuto mediante l'algoritmo Optimal Path.

Nei seguenti paragrafi viene descritto come funziona l'analisi di ogni segmento di percorso e l'eventuale correzione dei singoli tratti.

6.4.1 Analisi del primo tratto

Il primo tratto generato dall'algoritmo di Dubins è un arco di circonferenza che porta il robot a curvare in senso orario o antiorario. Questo primo tratto è caratterizzato da una lunghezza dell'arco t e da un raggio di curvatura $R_{2,min}$ riferito al modulo posteriore. Il valore iniziale dell'angolo $\delta_{\mathrm{I,i}}$ è noto in quanto la configurazione del robot all'inizio del percorso è una grandezza di input dell'algoritmo:

$$\delta_{\mathrm{I,i}} = \delta_{\mathrm{O}}$$

Noti i valori di t e di $R_{2,min}$ è possibile ricavare facilmente l'angolo θ_I sotteso dall'arco di circonferenza mediante la relazione:

$$\theta_I = \frac{t}{R_{2,min}}$$

Una rappresentazione di questi parametri è visibile in Figura 6.16.

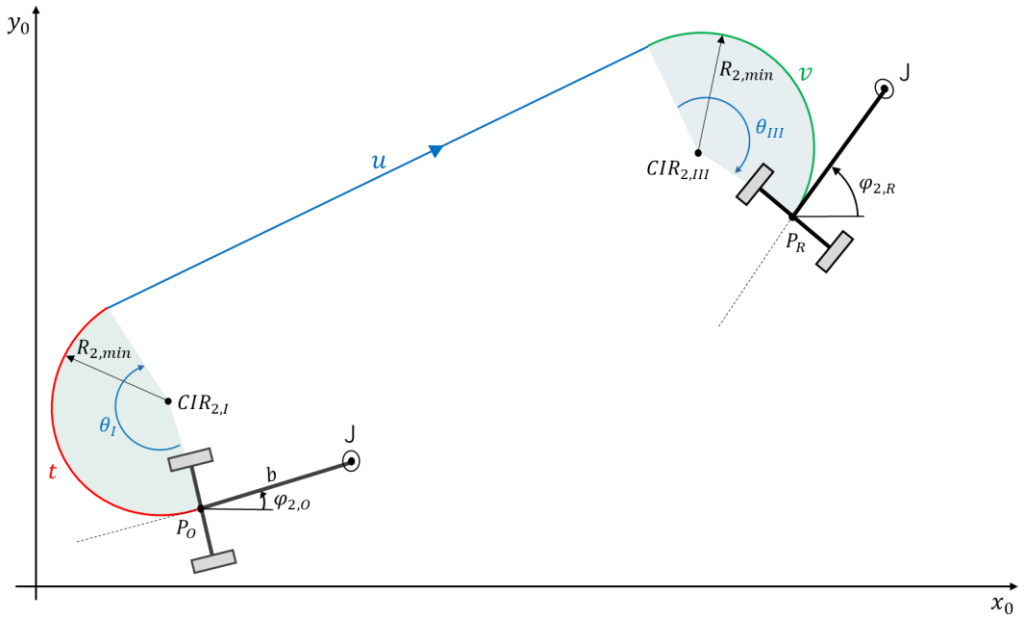

Figura 6.16 – Rappresentazione dei tre tratti del percorso di Dubins e relativi parametri

Note queste grandezze si applica la funzione di previsione $f_1(R_2, \theta_I, \delta_{\mathrm{I,i}})$ la quale restituisce, in uscita, il valore dell'angolo $\delta_{\mathrm{I,f}}$ assunto dal robot alla fine del primo tratto. Tale valore viene confrontato col valore massimo ammissibile $\delta_{\max}$:

- Se $|\delta_{\mathrm{I,f}}| < \delta_{\max}$ allora il primo tratto non porta il modulo posteriore a chiudersi su quello anteriore col rischio di collisione. Si può perciò procedere con l'analisi del tratto successivo.
- Se $|\delta_{\mathrm{I,f}}| \geq \delta_{\max}$ allora il primo tratto porterebbe al fenomeno del jackknife in retromarcia. In questo caso si aumenta il raggio minimo di curvatura $R_{2,min}$ e si ricalcola il percorso di Dubins tra la configurazione iniziale P_O e quella finale P_R. Si otterrà quindi un percorso Dubins totalmente nuovo e si riprenderà ad analizzare ogni tratto partendo dal primo.

6.4.2 Analisi del secondo tratto

Il secondo tratto, a differenza del primo e del terzo, può essere sia un arco di circonferenza che un tratto rettilineo.

- Se il secondo tratto è curvilineo allora sono noti la lunghezza u dell'arco ed il raggio di curvatura $R_{2,min}$. In modo analogo a quanto è stato fatto per il primo tratto si può ricavare l'angolo θ_{II} sotteso dall'arco di circonferenza.

$$\theta_{II} = \frac{u}{R_{2,min}}$$

  A questo punto è possibile applicare la funzione di previsione $f_1(R_2,\ \theta_{II},\ \delta_{\mathrm{II,i}})$.
- Nel caso in cui il secondo tratto corrisponda ad un tratto rettilineo è necessario applicare la funzione di previsione $f_2(u,\ \delta_{II,i})$ dove u rappresenta la lunghezza del tratto.

È importante notare come in entrambi i casi il valore dell'angolo δ all'inizio del secondo tratto corrisponda al valore di δ calcolato alla fine del primo tratto.

$$\delta_{II,i} = \delta_{I,f}$$

Applicando, a seconda dei casi, una delle due funzioni di previsione si ottiene un valore di δ alla fine del secondo tratto. Come per il primo tratto, tale valore viene confrontato con $\delta_{\max}$:

- Se $|\delta_{\mathrm{II,f}}| < \delta_{\max}$ allora il robot percorre il secondo tratto senza che si presenti il jackknifing. Si può quindi procedere con il controllo del terzo tratto.
- Se $|\delta_{\mathrm{II,f}}| \geq \delta_{\max}$ allora il secondo tratto porta il robot al fenomeno del jackknife. In questo caso si tiene memoria del primo tratto, cioè del percorso che porta dal punto P_0 al punto $P_{f,I}$, e si cancellano i tratti II e III. Dopodiché si ricalcola il percorso di Dubins tra la configurazione $P_{f,I}$ e la configurazione P_R. Anche in questo caso è prevista una maggiorazione del raggio di curvatura $R_{2,min}$. Alla fine di questo processo si otterranno quindi 4 tratti, il tratto I seguito dai tre tratti ottenuti dal calcolo del percorso di Dubins successivo alla correzione. A questo punto si provvede al controllo di tutti i tratti che si trovano dopo il primo.

6.4.3 Analisi del terzo tratto

Analogamente al primo tratto, noto il raggio $R_{2,min}$ e la lunghezza v dell'arco di circonferenza, è possibile ricavare l'angolo θ_{III} che sottende l'arco.

$$\theta_{III} = \frac{v}{R_{2,min}}$$

Inoltre, è possibile affermare che il valore di δ all'inizio del terzo tratto corrisponde al valore di δ alla fine del secondo tratto.

$$\delta_{III,i} = \delta_{II,f}$$

Noti questi elementi viene applicata la funzione di previsione $f_1(R_2, \theta_{III}, \delta_{III,i})$ ottenendo il valore di δ alla fine del terzo tratto. Si procede quindi al confronto tra l'angolo δ trovato e l'angolo $\delta_{\max}$:

- Se $|\delta_{III,f}| < \delta_{\max}$ allora il robot percorre il terzo tratto senza che vi sia jackknifing. Il percorso trovato è perciò approvato in quanto l'ultimo tratto, nonché nessuno dei tratti precedentemente analizzati, porta al fenomeno del jackknife.
- Se $|\delta_{III,f}| \geq \delta_{\max}$ allora il terzo tratto non può essere percorso dal robot. Si tiene quindi memoria del primo e del secondo tratto, cioè del percorso dal punto P_0 al punto $P_{f,II}$. Successivamente si ricalcola il percorso Dubins partendo dal punto $P_{f,II}$ al punto P_R applicando una maggiorazione del raggio di curvatura $R_{2,min}$. Alla fine di questo processo si otterranno 5 tratti, il tratto I e II seguiti dai tre tratti ottenuti dal calcolo del percorso Dubins successivo. A questo punto si provvede nuovamente al controllo dei nuovi tratti calcolati.

Al termine si ottiene il percorso cercato, formato da un certo numero di tratti, rettilinei o curvilinei, che permettono di passare da una configurazione iniziale in cui si trova il robot ad una configurazione finale riducendo notevolmente le probabilità del verificarsi del jackknife.

Nello specifico, il percorso che si ottiene in uscita dalla funzione Optimal Path, viene corredato da vettori e parametri che permettono di identificarne gli aspetti di maggior interesse:

- $path$: posa (x_2, y_2, φ_2) assunta dal modulo posteriore del robot per ogni punto del percorso;
- $original\_length$: lunghezza del percorso di Dubins di partenza;
- $length\_path$: vettore contenente la lunghezza cumulativa del percorso ad ogni posa;
- $rho2$: vettore contenete la curvatura ρ_2 (con segno) del percorso per ogni posa del modulo posteriore;
- $part\_length$: lunghezza di ogni tratto e tipo di tratto (1: tratto curvilineo, 2: tratto rettilineo). La lunghezza di questo vettore fornisce anche un'indicazione sul numero di tratti ottenuti;

Ovviamente, ogni volta che il percorso viene corretto aumentando il raggio di curvatura, si ha un aumento della lunghezza del percorso rispetto a quello originale. Se si considerasse la velocità longitudinale v_1 costante ciò vorrebbe dire che, passando dal percorso di Dubins al percorso ottimale, il robot impiegherebbe più tempo per eseguire la manovra in retromarcia. Tuttavia, se si considera un caso reale, ogni volta che un veicolo percorre una curva si riduce la sua velocità per mantenerlo sulla traiettoria; più la curva è stretta. maggiore deve essere la decelerazione. Quindi nel percorso ottimale le curve a raggio di curvatura maggiore possono essere percorse ad una velocità più alta rispetto alle curve strette ottenute dall'algoritmo di Dubins. Questo permette di compensare, uguagliare o addirittura ridurre i tempi di percorrenza del percorso.

6.4.4 Considerazioni sul raggio di curvatura minimo $R_{2,min}$ e sulla sua correzione

Come descritto nel Capitolo 2 i robot Agri.q ed Epi.q sono robot composti da moduli a guida differenziale[2], questo implica che il singolo modulo possa ruotare sul posto, cioè con raggio di curvatura nullo. Quindi in prima analisi si potrebbe indicare $R_{2,min} = 0\ m$. Tuttavia, l'algoritmo di Dubins utilizzato non permette di porre il raggio minimo pari a 0. Per evitare che questo aspetto vada a limitare il range di mobilità del robot, è possibile ovviare al problema inserendo come input dell'algoritmo un raggio di curvatura minimo estremamente piccolo, minore della metà dell'interasse i. In questo modo si permette al robot di compiere delle rotazioni in cui il centro di istantanea rotazione si trovi nel volume del robot stesso.

- Per Epi.q: $R_{2,min} < \frac{i}{2} = 0.13\ m = 13\ cm$
- Per Agri.q: $R_{2,min} < \frac{i}{2} = 0.42\ m = 42\ cm$

È necessario considerare anche un ulteriore aspetto: più il raggio di curvatura è piccolo più facilmente si verifica il fenomeno del jackknifing, non per niente l'algoritmo Optimal Path va ad aumentare il raggio di curvatura dei singoli tratti quanto riscontra un possibile

[2] Se si considera il robot Agri.q solo il modulo frontale è a guida differenziale.

urto tra i moduli[3]. Per questo, nella quasi totalità dei casi, l'algoritmo Optimal Path va a ridurre la curvatura ρ_2 dei tratti del percorso che presentano archi con curvature troppo accentuate; quindi, il range di mobilità che viene escluso ponendo $R_{2,min} > 0\ m$ verrebbe comunque corretto nella maggior parte dei casi per evitare il jackknife.

Ogni volta che l'algoritmo Optimal Path trova che il tratto analizzato porta ad avere un angolo δ maggiore del valore massimo, aumenta il raggio di curvatura del 30% rispetto al valore attuale. È stata scelta questa percentuale di maggiorazione in quanto da una parte permette di avere un aumento non eccessivo degli archi di circonferenza che costituiscono il percorso (e di conseguenza di non avere dei percorsi eccessivamente lunghi), dall'altra permette di non avere un numero troppo elevato di iterazioni per trovare il tratto di percorso idoneo, ciò permette di non avere tempi di calcolo della traiettoria eccessivamente lunghi.

Di seguito viene mostrato un confronto, a parità di configurazione iniziale e finale, tra un percorso ottenuto mediante l'algoritmo di Dubins ed un percorso ottenuto mediante l'algoritmo Optimal Path.

Robot: Epi.q
Tipo di moto: moto in retromarcia
Input:
$P_O = (0\ m, 0\ m, 0°)$
$P_R = (0\ m, 1\ m, 90°)$

Caratteristiche percorso di Dubins:
$R_{2,min} = \frac{i}{2} \cdot 0.7 = 0.091m$
$t = 0.13\ m$
$u = 0.91\ m$
$v = 0.29\ m$
$L_{tot} = 1.34\ m$
$T_{calcolo} = 0.002\ s$

Caratteristiche percorso ottimale:
$L_{tot} = 2.88\ m$
$T_{calcolo} = 0.01\ s$

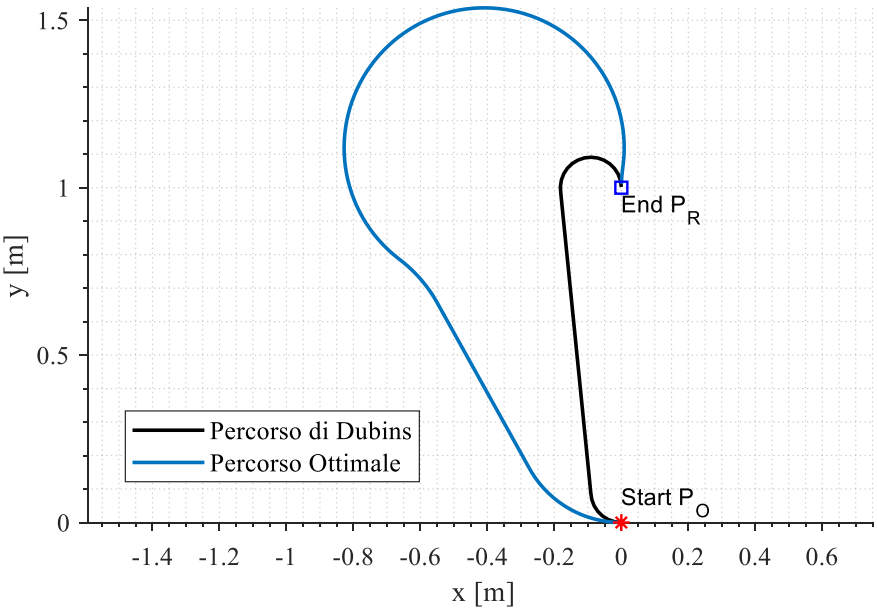

Figura 6.17 – Confronto tra percorso di Dubins e Optimal Path.

[3] Una eventuale rotazione sul posto, infatti, potrebbe essere eseguita solo per un certo arco di circonferenza; se la rotazione continuasse si arriverebbe inevitabilmente al contatto tra i due moduli. A quel punto sarebbe necessaria una rotazione nel senso opposto per poter riallineare i moduli del robot.

In questo caso cambia sia il numero che la lunghezza di ogni tratto, in quanto vi è stato un aumento del raggio di curvatura dei tratti curvilinei:

Tabella 6.2 – Caratteristiche dei tratti che compongono il percorso ottenuto mediante l'algoritmo Oprimal Path

Tratti	**Tipologia**	**Lunghezza [m]**	**Raggio di curvatura R_2** [m]
I	Arco di cerchio	0.3	- 0.31
II	Segmento rettilineo	0.8	inf
III	Arco di cerchio	0.17	0.42
IV	Arco di cerchio	1.6	- 0.42

Il percorso ottenuto viene poi corretto per rimuovere le discontinuità di curvatura.

Osservando questo esempio si può notare come l'algoritmo Optimal Path necessiti di un maggior tempo di calcolo. A seguito di diverse prove si può affermare che, passando dal percorso di Dubins all'algoritmo Optimal Path, il tempo di calcolo aumenta di un ordine di grandezza, passando dai centesimi di secondo ai decimi di secondo. Ciò vuol dire che Optimal Path resta comunque un algoritmo estremamente veloce.

6.4.5 Ottenere il robot allineato al termine del percorso

Un aspetto in più che è possibile considerare riguarda l'eventuale necessità di avere i due moduli allineati al termine del percorso, il che equivale al volere l'angolo $\delta = 0$ in corrispondenza della configurazione finale.

Un primo approccio potrebbe essere quello di raggiungere la posizione finale e successivamente applicare una rotazione sul posto del modulo anteriore per portare il robot nella configurazione con i due moduli allineati. Questa possibilità è però stata scartata in virtù delle seguenti considerazioni.

Nel caso di Epi.q, poiché possiede un braccio a non nullo, una rotazione del modulo anteriore causerebbe un'ulteriore rotazione del modulo posteriore, che non si troverebbe più orientato nel modo voluto. Quindi si finirebbe per avere il robot con i due moduli allineati, ma in una configurazione che si discosta da quella finale desiderata (Figure 6.18 e 6.19). Inoltre, l'idea di creare manovre "a pezzi" (alternanza di sterzate ed accelerazioni) comporterebbe notevoli fenomeni dinamici; quindi, per realizzarla, servirebbe uno specifico

inseguitore di traiettoria in modo da compensare gli spostamenti non desiderati di uno dei due moduli durante la rotazione sul posto. Questo è anche il motivo per cui l'algoritmo Optimal Path corregge i tratti del percorso aumentando il raggio di curvatura e non applicando una rotazione sul posto tra un tratto e l'altro.

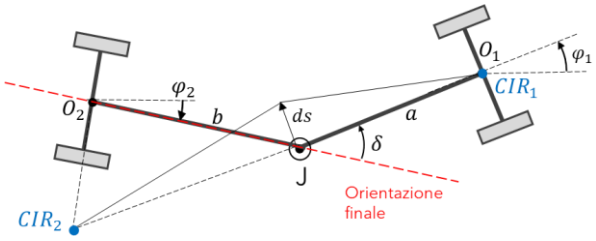

Figura 6.18 – Rotazione sul posto del modulo anteriore di Epi.q

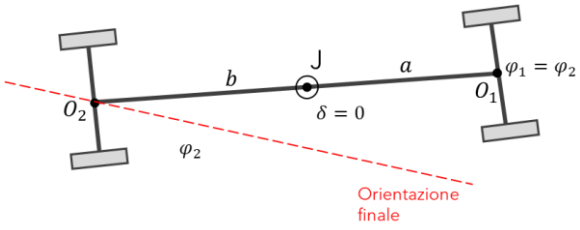

Figura 6.19 – I moduli del robot Epi.q allineati

Questa soluzione è invece applicabile sul robot Agri.q perché una rotazione del modulo anteriore non si ripercuote sul modulo posteriore poiché la lunghezza a è nulla (Figure 6.20 e 6.21). Benchè da un punto di vista cinematico questo approccio sia la soluzione da preferire, in molti casi pratici si è potuto constatare che il robot Agri.q, su certi terreni come lo sterrato, non è in grado di eseguire una rotazione sul posto del modulo frontale.

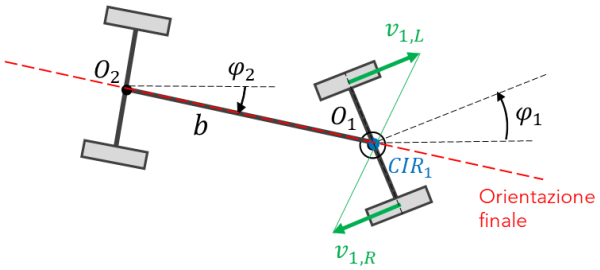

Figura 6.20 – Rotazione sul posto del modulo anteriore di Agri.q

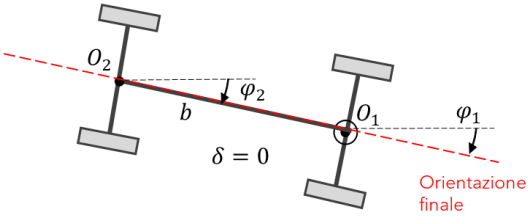

Figura 6.21 – I moduli del robot Agri.q allineati

Per queste ragioni è stato adottato un altro tipo di approccio, che viene riportato di seguito.

È possibile inserire nel percorso un tratto rettilineo finale di lunghezza tale da ottenere i moduli allineati al termine della traiettoria. In quest'ottica la pianificazione del percorso non viene impostata rispetto al punto finale, ma rispetto ad un punto intermedio allineato al punto finale, dove la direzione della congiungente tra questi due punti corrisponde alla direzione di allineamento corretto, cioè la direzione della congiungente corrisponde

all'orientamento del modulo posteriore che si vuole ottenere al termine del percorso (φ_{2R}). Ciò equivale a dire che il percorso di Dubins non viene più calcolato tra la configurazione iniziale P_O e finale P_R, bensì tra il punto iniziale P_O ed il punto intermedio P_M. Di conseguenza anche l'algoritmo Oprimal Path eseguirà l'analisi dei tratti del percorso di Dubins tra P_O e P_M.

In questo modo, una volta raggiunto P_M, per arrivare alla configurazione finale al percorso Optimal Path calcolato tra P_O e P_M basterà aggiungere un tratto rettilineo che vada ad unire i punti P_M e P_R (Figura 6.22).

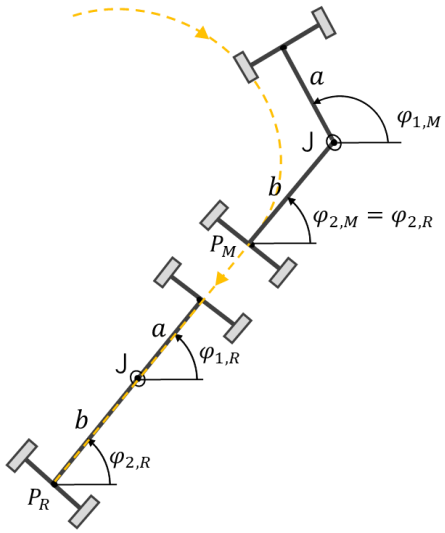

Figura 6.22 – Rappresentazione schematica dell'andamento di Epi.q lungo il tratto rettilineo finale

Nota la configurazione finale P_R (ed in particolare l'angolo φ_{2R} del modulo posteriore al termine del percorso), si procede col determinare il punto intermedio P_M. Si immagina di tracciare una circonferenza con centro corrispondente al punto P_R. Il raggio q di tale circonferenza equivale alla distanza che si vuole far percorrere al robot in marcia rettilinea in modo da raddrizzarsi (Figura 6.23). Sperimentalmente è stato visto che un buon valore di q corrisponde a 3 volte l'interasse i delle ruote anteriori. Tale valore permette di ottenere un allineamento finale anche in caso di moduli ad inizio rettilineo molto ripiegati tra loro (angolo δ elevato ad inizio rettilineo). È possibile affermare che P_M è un punto appartenente a questa circonferenza.

Per capire quale, tra i punti che costituiscono la circonferenza sia il punto intermedio cercato, si traccia una retta passante per il punto P_R e con angolo di orientamento pari a φ_{2R} (il quale in questo caso corrisponde a φ_{1R} poiché al termine del percorso si vogliono i due

moduli allineati). Dall'intersezione tra retta e circonferenza si ottengono due punti. Prendendo in considerazione il moto in marcia indietro si può determinare quale dei due punti corrisponda al punto P_M voluto.

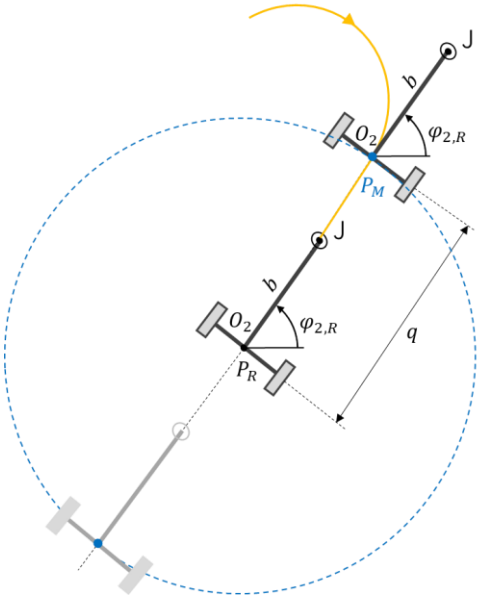

Figura 6.23 – Definizione del punto intermedio P_M

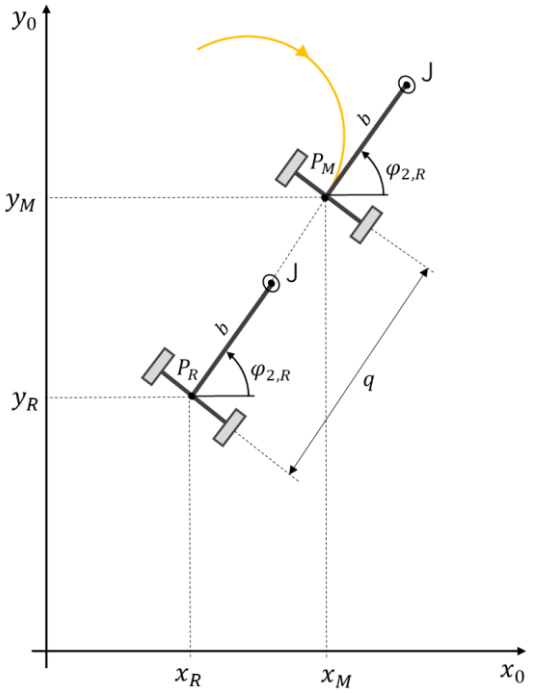

Figura 6.24 – Calcolo delle coordinate del punto intermedio P_M

Partendo dalla definizione del punto P_M si ricavano le corrispondenti coordinate x_M e y_M (Figura 6.24).

$$x_M = x_R + q \cos \varphi_{2,R}$$

$$y_M = y_R + q \sin \varphi_{2,R}$$

Un approccio migliore potrebbe consistere nell'adattare la lunghezza del tratto rettilineo finale all'angolo δ che si ottiene in corrispondenza del punto P_M: maggiore è il valore di δ più lungo dovrà essere il tratto per permettere ai due moduli di allinearsi. Si lascia questo aspetto come sviluppo futuro.

Si seguito viene riportato un esempio.

Robot: Epi.q

Tipo di moto: moto in retromarcia

Input:
$P_O = (0\ m, 0\ m, -90°)$
$P_R = (-2\ m, 1\ m, -45°)$

Caratteristiche percorso:
$L_{tot,} = 2.30\ m$
$T_{calcolo} = 0.023\ s$

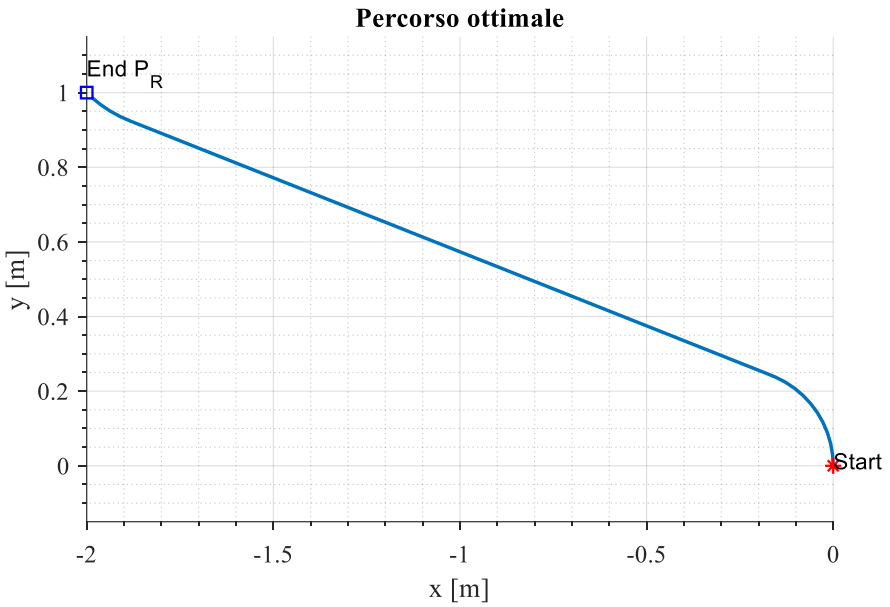

Figura 6.25 – Percorso in retromarcia

Robot: Epi.q

Tipo di moto: moto in retromarcia

Input:
$P_O = (0\ m, 0\ m, -90°)$
$P_R = (-2\ m, 1\ m, -45°)$
$\delta_R = 0°$
$q = 0.78\ m$

Caratteristiche percorso:
$L_{tot} = 2.56\ m$
$T_{calcolo} = 0.031\ s$

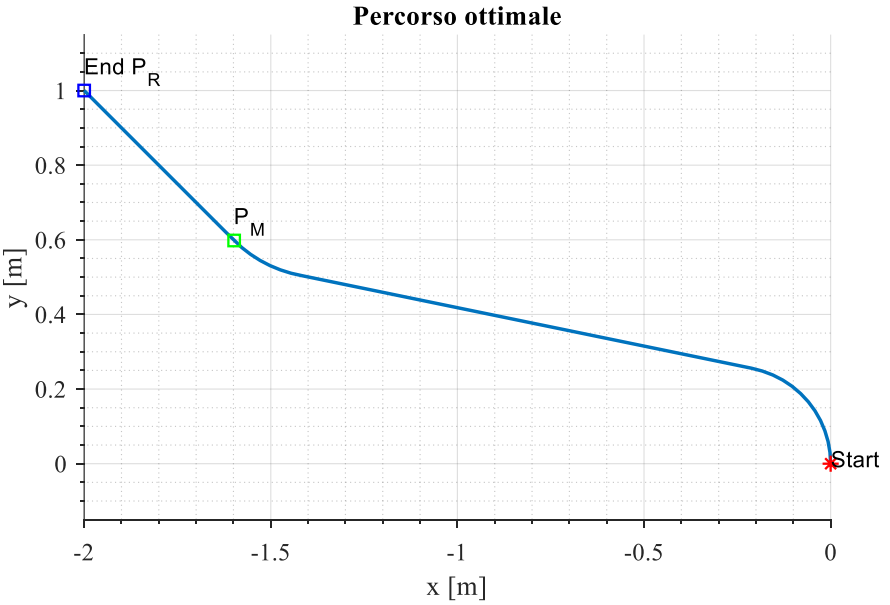

Figura 6.26 – Percorso in retromarcia tale da avere i moduli allineati al termine

Tabella 6.3 – Caratteristiche dei tratti che compongono il percorso ottenuto mediante l'algoritmo Oprimal Path

Tratti	Tipologia	Lunghezza [m]	Raggio di curvatura R_2 [m]
I	Arco di cerchio	0.35	0.26
II	Segmento rettilineo	1.24	Inf
III	Arco di cerchio	0.19	-0.34
IV	Segmento rettilineo	0.78	inf

Il percorso viene successivamente corretto in modo da rimuovere le discontinuità di curvatura.

7. Capitolo 7: Pianificazione di traiettoria in presenza di ostacoli

7.1 Ingombro del robot

Il primo passo per poter controllare il robot nel caso di un ambiente in cui sono presenti ostacoli è quello di definire l'ingombro del robot. Quest'analisi si rende necessaria perché, in alcuni casi, anche se gli algoritmi di esplorazione trovano un percorso che permette di evitare gli ostacoli, non è detto che il robot sia in grado di realizzarlo; ad esempio, tratti rettilinei tra pareti molto vicine non possono essere eseguiti se l'ingombro del robot è maggiore del corridoio che viene a formarsi tra le pareti.

Posto che il percorso da seguire sia già noto, un modo semplice per visualizzare graficamente l'ingombro del robot è quello di aggiungere altre due linee distanti s rispetto alla linea che definisce il percorso del modulo posteriore, una a destra ed una a sinistra, come mostrato in Figura 7.1.

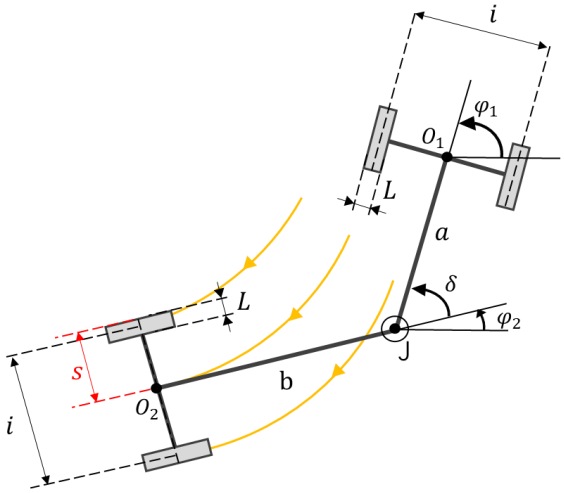

Figura 7.1 – Rappresentazione schematica del robot Epi.q e della grandezza s

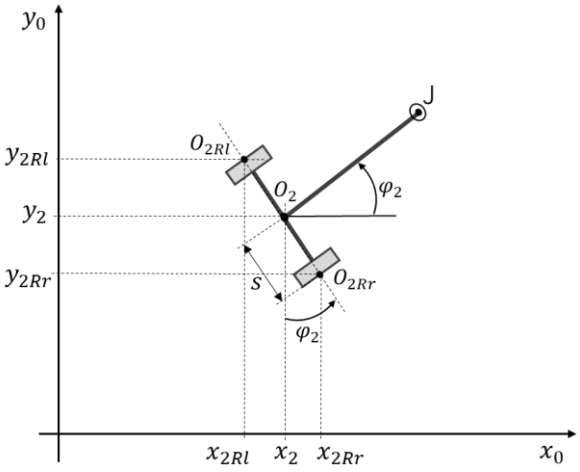

Figura 7.2 – Rappresentazione grafica delle coordinate dei due percorsi

Le coordinate dei due margini che rappresentano l'ingombro del robot, paralleli al percorso seguito dal punto O_2, vengono calcolate nel modo seguente (Figura 7.2):

$$\begin{cases} x_{2Rl} = x_2 - s\sin\varphi_2 \\ y_{2Rl} = y_2 + s\cos\varphi_2 \end{cases}$$

$$\begin{cases} x_{2Rr} = x_2 + s\sin\varphi_2 \\ y_{2Rr} = y_2 - s\cos\varphi_2 \end{cases}$$

Per definire la grandezza s è necessario considerare la geometria del robot. Osservando la Figura 7.1 si può dedurre che:

$$s = \frac{i + L}{2}$$

In questo modo è possibile visualizzare non solo il percorso per andare da una configurazione ad un'altra, ma anche avere un'idea dell'area occupata dal robot lungo tale percorso.

Vengono riportati alcuni esempi.

Robot: Epi.q
$P_O = (0\ m, 0\ m, -90°)$
$P_R = (2\ m, -1\ m, 120°)$
$L_{tot} = 2.64\ m$
$s = 0.155\ m$

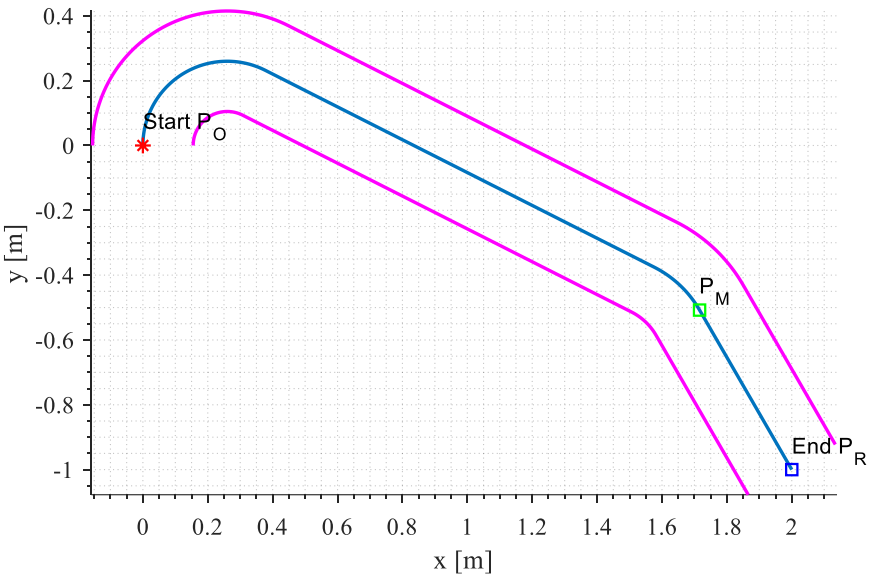

Figura 7.3 – Percorso in retromarcia (in blu), ingombro del modulo posteriore di Epi.q lungo tale percorso (in viola)

Robot: Agri.q
$P_O = (0\ m, 0\ m, 0°)$
$P_R = (2\ m, -1\ m, 120°)$
$L_{tot} = 16.80\ m$
$s = 0.448\ m$

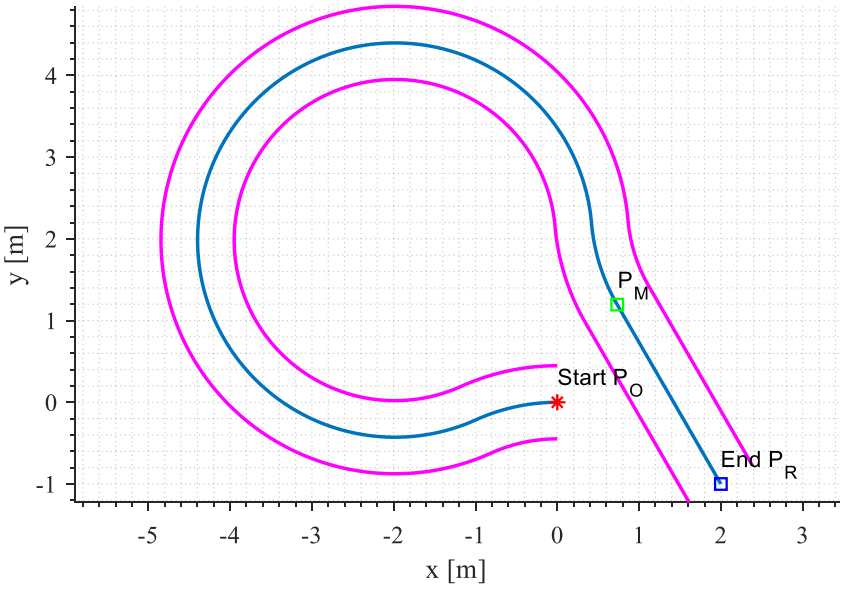

Figura 7.4 – Percorso in retromarcia (in blu), ingombro del modulo posteriore di Agri.q lungo tale percorso (in viola)

7.2 Percorso per collegare una serie di waypoints

Una volta definito l'algoritmo Optimal Path per portare il robot da una configurazione ad un'altra, è possibile definire un percorso in retromarcia, passante per una serie ordinata di punti, applicando man mano questo algoritmo tra coppie di punti successivi.

Da tale contesto scaturiscono due riflessioni:

- Nella maggior parte dei casi risulta necessario ottenere l'allineamento dei moduli che compongono il robot solo in corrispondenza dell'ultimo punto, mentre nei punti intermedi l'allineamento non è richiesto. Questo vuol dire che il tratto rettilineo per ottenere $\delta = 0$ va aggiunto, eventualmente, solo al termine del percorso complessivo eseguito in retromarcia.
- Non sempre l'orientamento φ_2 del modulo posteriore del robot in un dato waypoint è fondamentale, l'interesse può essere solo quello che il robot passi su un determinato punto del percorso, senza dover assumere uno specifico angolo φ_2.

Uno dei limiti dell'algoritmo Optimal Path è che non è possibile fornire a quest'ultimo due configurazioni di input senza indicare l'orientamento φ_2 del modulo posteriore. Un modo semplice per aggirare tale problema e allo stesso tempo avere in uscita il percorso più semplice possibile (dove per semplice si intende col minor numero di curve non necessarie) viene di seguito discusso.

Noto un generico punto $P_A = (x_{2A}, y_{2A}, \varphi_{2A})$ del percorso, si traccia il segmento che unisce tale punto a quello successivo (P_B) e si suppone di voler ottenere il robot allineato con tale segmento in corrispondenza del punto P_B (Figura 7.5). Nota la posizione di P_A e P_B è possibile ricavare l'angolo φ_{2B} e quindi la configurazione che si vuole ottenere in corrispondenza di $P_B = (x_{2B}, y_{2B}, \varphi_{2B})$.

Al fine di determinare l'angolo φ_{2B} si sfrutta la funzione atan2 presente in Matlab:

$$\Delta x = x_{2B} - x_{2A}$$

$$\Delta y = y_{2B} - y_{2A}$$

$$\beta = atan2(\Delta y, \Delta x)$$

$$\varphi_{2B} = \pi + \beta$$

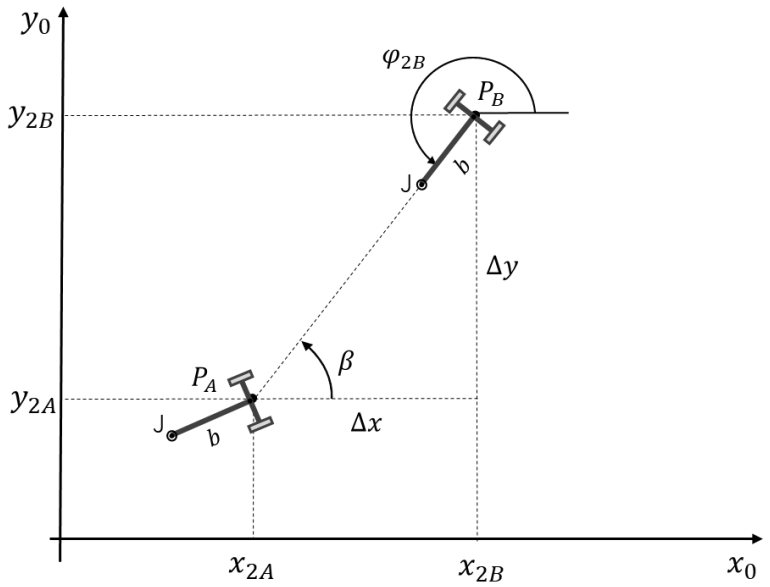

Figura 7.5 – Calcolo dell'angolo φ_{2B}

7.3 Definizione della mappa e degli ostacoli

Per poter studiare un algoritmo per il calcolo di un percorso in presenza di ostacoli è fondamentale conoscere la mappa, ossia lo spazio di lavoro in cui il robot mobile dovrà muoversi.

Si procede quindi col definire la mappa contenente gli eventuali ostacoli che il robot dovrà evitare. Per creare tale mappa la scelta è ricaduta sulla funzione `makemap.m` presente nel Robotics Toolbox di Peter Corke. Questa funzione permette di generare una mappa NxN sopra la quale si possono tracciare, trascinando il mouse, ostacoli di diverse dimensioni.

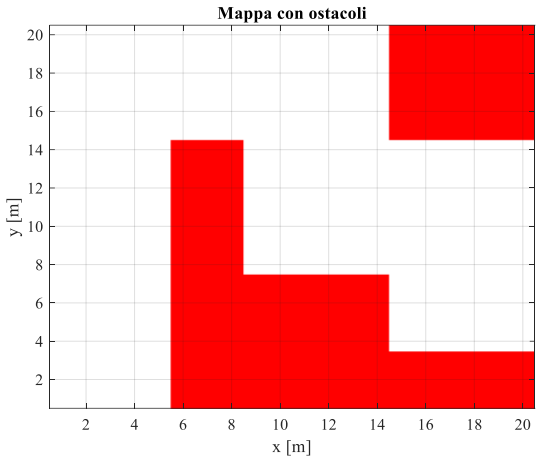

Figura 7.6 – Mappa 20x20 m con ostacoli (in rosso)

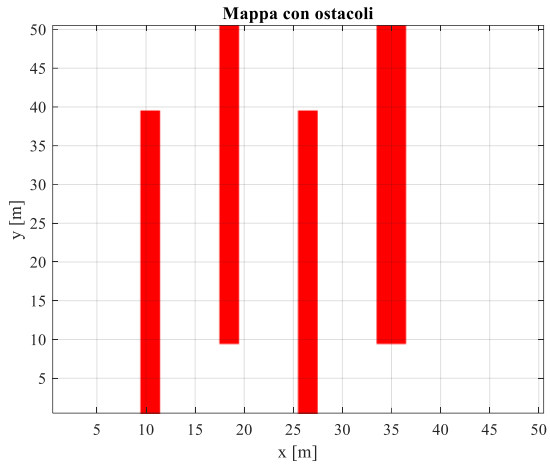

Figura 7.7 – Mappa 50x50 m con ostacoli (in rosso)

7.4 Algoritmo di navigazione PRM

Una volta che gli ostacoli, il punto di partenza ed il punto di arrivo sono stati definiti, è necessario determinare una serie di punti che portino dalla configurazione iniziale a quella finale evitando gli ostacoli. Questo obiettivo è raggiunto sfruttando gli algoritmi di navigazione, noti anche come funzioni di pianificazione.

Le funzioni di pianificazione del movimento sono algoritmi in grado di creare automaticamente percorsi privi di collisioni all'interno dell'area di lavoro del robot. Scegliere un algoritmo appropriato è particolarmente importante quando si parla di motion planning. I differenti algoritmi esistenti in letteratura, infatti, sono stati sviluppati per particolari situazioni, principalmente legate alle dimensioni dello spazio libero.

Oggi sono disponibili diversi algoritmi di pianificazione del movimento per la robotica, in particolare esistono due grandi famiglie di algoritmi che si basano sul campionamento dello spazio delle configurazioni [41]. Si parla di algoritmi di tipo Single-Query Planning quando, dato un determinato ambiente virtuale, è necessario attuare un'unica pianificazione. Se invece all'interno dello stesso ambiente virtuale vengono richieste diverse pianificazioni si parla di Multiple-Query Planning. La differenza fondamentale tra questi due tipi di approcci risiede nel fatto che le tecniche di Multiple-Query Planning effettuano un pre-processing dello spazio delle configurazioni per velocizzare la fase successiva di pianificazione vera e propria. Il pre-processing ha tempi computazionali rilevanti, per questo motivo non è

conveniente utilizzare tali tecniche nel caso in cui si debbano effettuare poche pianificazioni. La più vasta famiglia di algoritmi di tipo multiple-query sono le Probabilistic Road Maps (PRM), mentre i Rapidly-Exploring Random Trees (RRTs) vengono preferiti se si utilizzano tecniche single-query.

La Probabilistic Road Maps (PRM) è un tipo molto comune di algoritmo di pianificazione del movimento 'Collision-Free'. Il termine *probabilistico* è dovuto al fatto che l'insieme di punti che il pianificatore usa per sapere dove il robot può muoversi nel suo spazio di lavoro sono scelti casualmente. Questo posizionamento casuale è chiamato *probabilistico*. L'alternativa è un posizionamento *uniforme*, in cui i punti vengono posizionati a intervalli regolari nell'intero spazio di lavoro. Questo rende gli algoritmi PRM più veloci di altri algoritmi di pianificazione del movimento che cercano di coprire equamente l'intero spazio di lavoro.

A differenza di alcuni pianificatori di movimento, inoltre, gli algoritmi PRM dividono il loro lavoro in due fasi che vengono eseguite separatamente. In primo luogo, gli algoritmi PRM creano una Roadmap, cioè una mappa dello spazio libero nello spazio di lavoro del robot. Può essere necessario un po' di tempo per generare questa mappa, ma una volta che è stata creata non è necessario calcolarla di nuovo in modo da poter "interrogare" la mappa molto più velocemente. Questa mappa viene poi utilizzata per generare rapidamente traiettorie libere da collisioni durante il funzionamento del robot.

Di seguito vengono descritte più nel dettaglio le due fasi.

7.4.1 Fase di Learning o di costruzione

La prima fase della pianificazione consiste nel generare la Roadmap, ovvero la rete di nodi e segmenti in cui i nodi rappresentano le diverse configurazioni attuabili dal robot e appartenenti allo spazio libero, mentre i segmenti sono le traiettorie realizzabili che li connettono.

L'algoritmo PRM genera la tabella di marcia delle località raggiungibili utilizzando i seguenti parametri:

- Numero di campioni: la mappa consiste in un numero di campioni (nodi) posizionati casualmente in tutto lo spazio di lavoro raggiungibile dal robot. Questo numero determina il numero di campioni.
- Archi: l'algoritmo tenta di collegare ogni campione nella mappa con un certo numero di altri campioni. La linea che collega due campioni insieme è chiamata 'arco' ed indica un percorso privo di collisioni tra queste due posizioni. Questo parametro può essere espresso come il numero massimo di archi per campione o andando a definire la lunghezza massima sotto la quale si considera possibile unire due campioni.

Questi parametri determinano quanto sarà dettagliata la Roadmap generata. Essi influenzano anche il tempo necessario per generare la Roadmap durante la fase di costruzione.

Durante questa fase il pianificatore posiziona un campione (nodo casuale) all'interno dello spazio di lavoro libero del robot. L'algoritmo verifica quindi il percorso tra questo campione e i campioni circostanti. Se il percorso tra i due nodi è libero da collisioni, aggiunge il percorso come 'arco' alla mappatura. Quando il pianificatore raggiunge il numero massimo di segmenti per questo campione, passa al campione successivo. In questo modo, al termine della fase di Learning, si ha a disposizione una tabella di marcia completa dei percorsi privi di collisioni all'interno dell'area di lavoro.

Questa fase richiede molto tempo, è la fase più lenta in quanto l'algoritmo ha molti campioni da testare. Tuttavia, deve essere eseguita una sola volta perché, una volta creata la mappa, non è necessario calcolarla nuovamente (a meno che non si aggiungano nuovi oggetti all'ambiente).

7.4.2 Fase di Quering o di interrogazione

Si forniscono al pianificatore due obiettivi (un punto di partenza ed un punto di arrivo). L'algoritmo cerca di trovare il percorso più breve e privo di collisioni tra i due punti obiettivo utilizzando la tabella di marcia creata nella fase di costruzione. In questo modo la pianificazione consiste solamente nella ricostruzione del percorso che unisce partenza e destinazione. Per questo la fase di interrogazione è particolarmente veloce e può essere ripetuta più volte.

7.4.3 Considerazioni sugli algoritmi PRM

Gli algoritmi PRM sono caratterizzati dalle seguenti proprietà [41, 42]:

- sono ideali in caso di richieste multiple di pianificazione perché, una volta costruita la rete di traiettorie nella prima fase, la pianificazione è velocissima;
- sono semplici e facili da implementare tramite Matlab;
- sono completi in senso probabilistico: la probabilità che un pianificatore trovi un cammino libero, se questo esiste, tende ad 1 al crescere del tempo di esecuzione (al tendere del tempo di esecuzione a infinito), oppure, all'aumentare dei punti, la probabilità di non trovare il percorso va a 0.

Si è trovato che, per ambienti particolarmente complessi (labirintici), l'algoritmo PRM trova la soluzione in tempi più brevi rispetto ad altri algoritmi come RRT [41]; ciò perché PRM genera i nodi coprendo con probabilità uniforme tutti gli spazi liberi e, dopo averli connessi, trova (se possibile) la traiettoria finale. Per ambienti meno complessi invece, risulta più conveniente RRT, poiché in questi casi non è necessario esplorare l'intero spazio delle configurazioni. In sostanza, l'algoritmo RRT, a parità del numero di nodi, è meno efficiente in termini di copertura dello spazio libero.

Inoltre, bisogna anche sottolineare che, in questo caso di studio, l'algoritmo PRM risulta particolarmente azzeccato poiché, creando una rete di nodi valida per qualunque punto in cui si trovi il robot, ben si adatta ad ambienti strutturati in cui potrebbe trovarsi a lavorare Agri.q. In tal senso si preferisce creare la tabella di marcia come una fitta rete di nodi, in quanto, se l'ambiente non varia, resta sempre valida, quindi, una volta calcolati tutti i nodi, non deve essere ricalcolata.

7.4.4 Applicazione dell'algoritmo PRM

Come per la definizione della mappa, anche in questo caso è stata scelta una funzione dalla libreria Robotics Toolbox. Le due fasi descritte precedentemente vengono realizzate mediante funzioni distinte. La fase di Learning è realizzata mediante la funzione `PRM.m` il cui funzionamento viene descritto di seguito.

```
prm = PRM(map,'npoints', npoints,'distthresh',1);
```

Osservando il codice si può vedere che la funzione `PRM.m` riceve in ingresso 3 elementi:

- La mappa NxN con gli eventuali ostacoli definita grazie alla funzione `makemap.m`. La funzione `PRM.m` presente nel Robotics Toolbox è stata scelta anche perché queste due funzioni sono state implementate per lavorare in sinergia tra loro.
- Il valore 'npoints', ossia il numero di campion in cui viene suddivisa la mappa.
- Il valore 'distthresh', cioè la massima distanza con cui si possono connettere due campioni (la massima lunghezza dell'arco). La distanza tra un punto del percorso e quello successivo dovrà essere minore di tale termine.

Dati questi tre elementi la funzione dà come output l'oggetto `prm`. Applicando a tale oggetto il metodo `plan` si ottiene la Roadmap.

```
prm.plan();
```

La seconda fase viene realizzata applicando il metodo `query` ed indicando il punto iniziale (start) in cui si trova il robot ed il punto finale che si vuole raggiungere (goal). Questi due punti non sono altro che le coordinate x e y delle configurazioni P_O e P_R.

```
points = prm.query(start, goal);
```

Al termine di questo processo si ottiene una serie di punti che permette di passare dallo start al goal evitando collisioni con gli ostacoli (un esempio è visibile in Figura 7.8).

È importante notare che l'algoritmo PRM specifica per ogni punto la posizione che questo assume nella mappa, ma non viene data alcuna indicazione sull'orientazione che deve assumere il robot su tale punto. Per definire la configurazione migliore per il robot in ogni punto si rimanda alla Sezione 7.2.

$Fase\ di\ Leaning\ =\ 0.5\ s$

$Start\ =(3\ m, 1.5\ m)$

$Goal\ =(8\ m, 8.5\ m)$

$Fase\ di\ Quering\ =\ 0.02\ s$

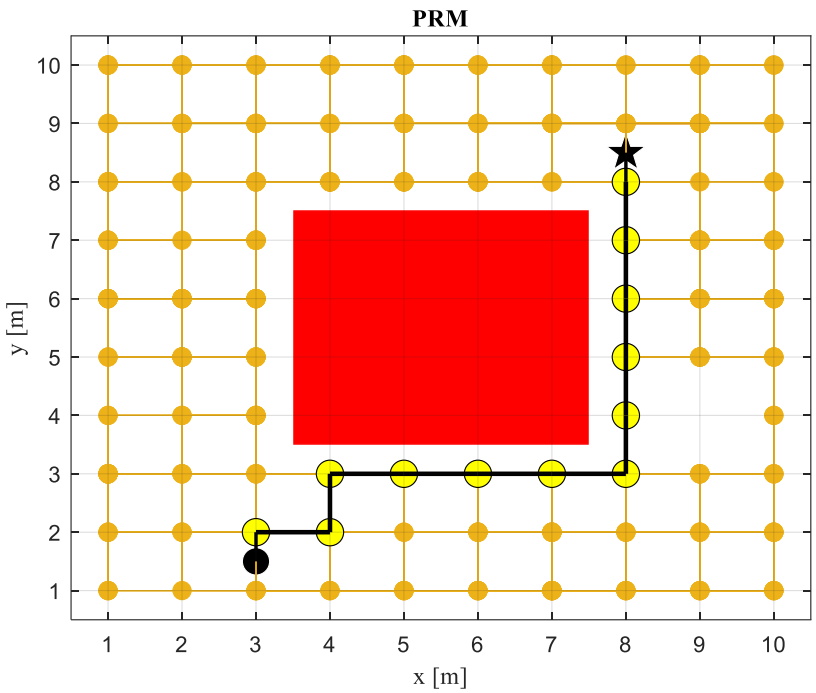

Figura 7.8 – Applicazione dell'algoritmo PRM in presenza di ostacoli

In Figura 7.8 viene riportato un esempio di ciò che si ottiene mendiate l'algoritmo PRM. Attraverso la prima fase è possibile ottenere la rete di punti rappresentata in arancione. Per generare questa rete si sono operate le seguenti scelte:

- Il numero di campioni (nodi) è stato posto pari alla dimensione della mappa (in questo caso = 10) per il numero di nodi occupati dagli ostacoli.
- Per quanto riguarda la lunghezza massima tra i nodi è stato deciso di inserire il valore più basso possibile, pari a 1 m, in modo da ottenere una rete fitta e dettagliata.

Mediante la seconda fase si ottiene la serie di punti gialli che permettono di andare dallo start (cerchio) al goal (stella) evitando l'ostacolo (in rosso).

Per comprendere il funzionamento dell'algoritmo successivo si evidenzia come non solo i vari nodi, ma anche gli ostacoli, siano espressi come una serie di punti sulla mappa, dove ogni punto è caratterizzato da una coordinata x e una coordinata y definite rispetto all'origine della mappa.

7.5 Interpolazione dei punti

Dopo aver definito la serie di punti che permette di tracciare un itinerario per raggiungere un punto preciso evitando gli ostacoli bisogna soffermarsi su come effettuare l'interpolazione tra un punto ed un altro.

La definizione del tragitto per evitare gli ostacoli può generare un certo numero di punti allineati tra loro disposti su un tratto rettilineo (come visibile in Figura 7.8). Ciò accade specialmente se la densità dei nodi è elevata. In questi casi non è conveniente fare l'interpolazione, sfruttando l'algoritmo di Dubins, tra ogni coppia di punti, in quanto potrebbe portare alla definizione di un percorso formato da inutili sinuosità quando sarebbe sufficiente un percorso rettilineo. Per evitare questo fenomeno si inizia applicando l'interpolazione col percorso di Dubins tra il primo e l'ultimo punto (tra P_O e P_R), se ciò non è realizzabile, perché il percorso che si ottiene va ad intersecare un ostacolo, allora si calcola il percorso Dubins tra il primo punto ed il penultimo e così via, finché non si riesce ad avere un primo percorso parziale che porta dalla posizione attuale del robot ad un punto N del tragitto senza incappare in un ostacolo. In altre parole, si calcola il percorso che permette al robot di passare dalla configurazione attuale al nodo N più lontano possibile, senza urtare contro un ostacolo. Ciò permette di utilizzare il percorso di Dubins evitando curve inutili.

Tale logica viene spiegata con maggior dettaglio nello schema in Figura 7.9.

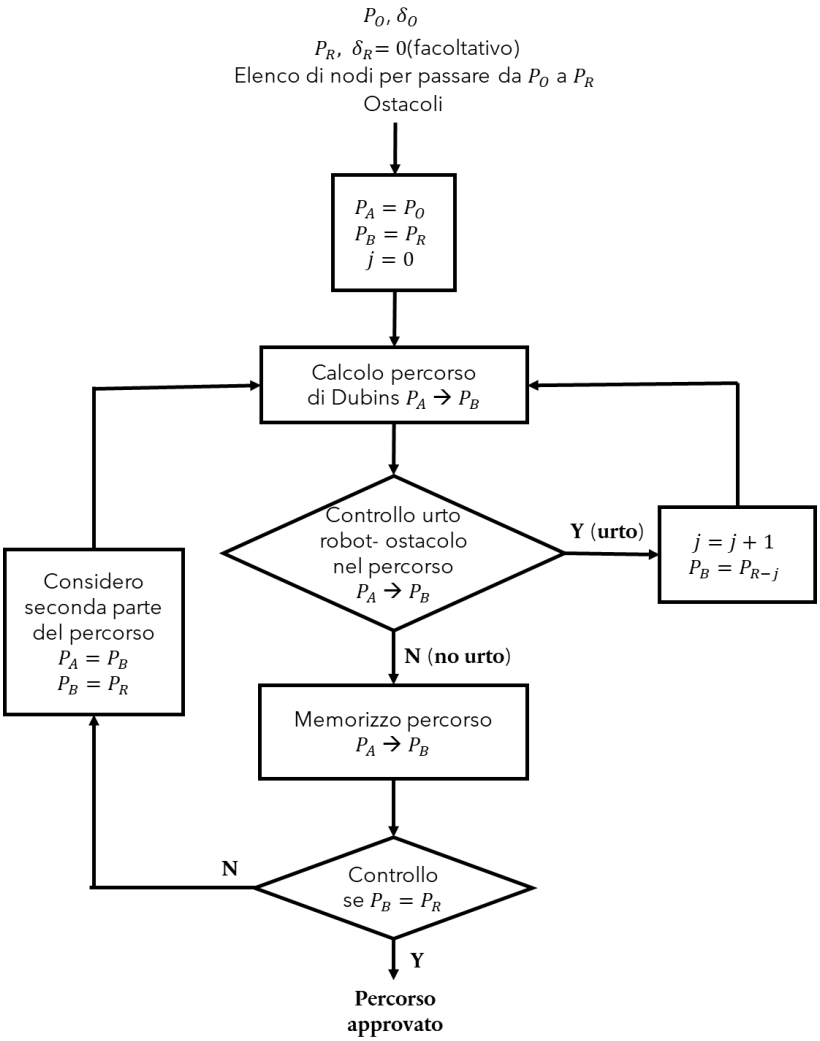

Figura 7.9 – Schema che descrive la logica con cui vengono interpolati i punti mediante il percorso di Dubins

Nell'esempio seguente si può vedere la differenza tra un percorso realizzato applicando l'interpolazione tra ogni coppia di punti ed un percorso ottenuto sfruttando la logica descritta.

$Fase\ di\ Leaning\ = 4.8\ s$

$Start\ = (2.3\ m, 4\ m)$

$Goal\ = (18\ m, 18.2\ m)$

$Fase\ di\ Quering\ = 0.22\ s$

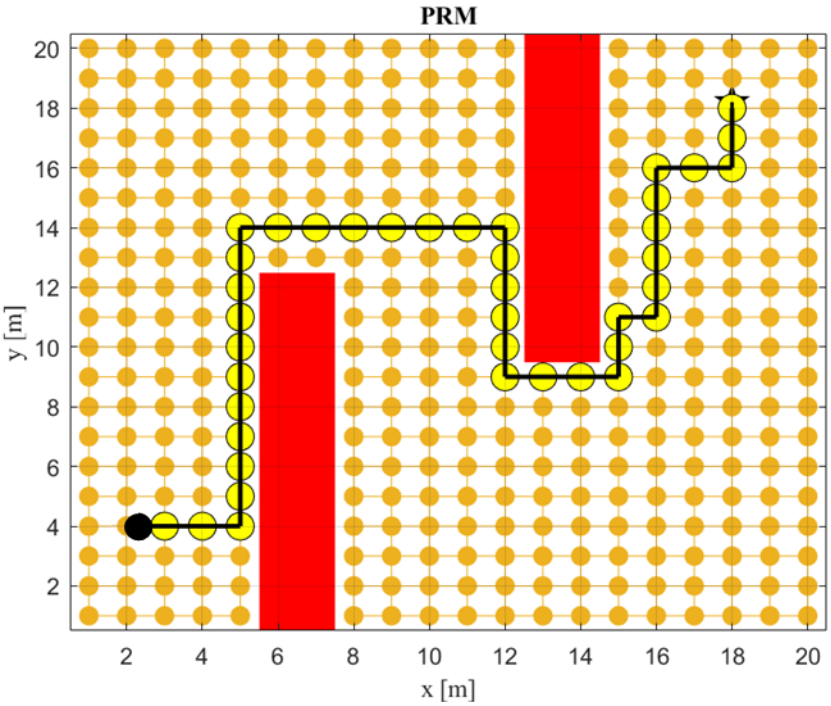

Figura 7.10 – Sequenza di punti ottenuta mediante l'algoritmo PRM

Robot: Epi.q

$P_O = (2.3\ m, 4\ m, -90°)$

$P_R = (18\ m, 18.2\ m, -90°)$

Caratteristiche percorso:

$L_{tot}\ = 42.5\ m$

$R_2 = 0.2\ m$

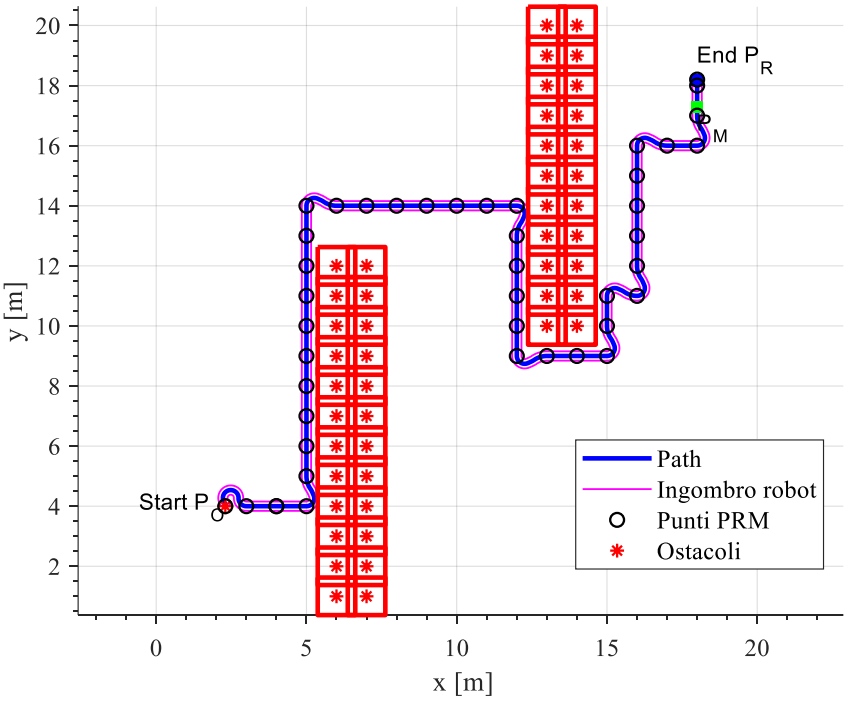

Figura 7.11 – Algoritmo di Dubins applicato tra ogni coppia di punti

Robot: Epi.q

$P_O = (2.3\ m, 4\ m, -90°)$

$P_R = (18\ m, 18.2\ m, -90°)$

Caratteristiche percorso:

$L_{tot} = 34.4\ m$

$R_2 = 1.7\ m$

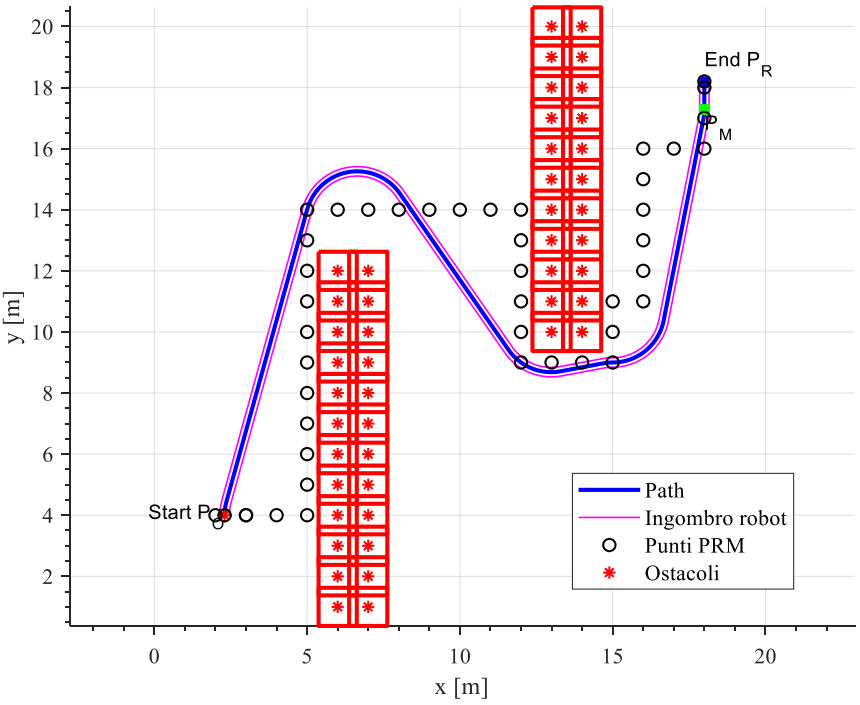

Figura 7.12 – Percorso ottenuto utilizzando la logica di interpolazione descritta

Come si può osservare dal confronto tra le Figure 7.11 e 7.12, utilizzando la logica di interpolazione (ed associandola ad un raggio di curvatura R_2 sufficientemente grande) è possibile ottenere un percorso caratterizzato da:

- Tratti rettilinei più lunghi.
- Raggi di curvatura maggiori.

Questi due aspetti portano ai seguenti vantaggi:

- È possibile affrontare le curve ed i tratti rettilinei a velocità maggiore (rispetto al caso in Figura 7.11) quindi si riducono i tempi di percorrenza del percorso in retromarcia.
- Avendo curve più ampie l'intervento del controllo sul moto in retromarcia è ridotto.
- Avendo curve più ampie il robot riesce più facilmente a seguire il percorso senza discostarsene in maniera eccessiva; quindi, si riduce il rischio di un contatto robot-ostacolo con previsto.

7.5.1 Perché la logica di interpolazione sfrutta l'algoritmo di Dubins

Nel Capitolo 6 si poteva notare come il passaggio dal percorso di Dubins all'Optimal Path portasse ad un aumento della lunghezza del percorso. Date le stesse configurazioni iniziali e finali, il percorso Optimal Path è in genere più lungo del percorso di Dubins. Ciò implica un aumento ''dell'ingombro'' del robot nel piano euclideo in quanto il luogo dei punti occupati dal robot è maggiore nel caso di percorso Optimal Path.

Questo vuol dire che la generazione di grandi archi di circonferenza al fine di evitare il fenomeno del jackknife mal si concilia con la necessità di evitare un ostacolo: più il percorso ricopre una vasta aera del piano maggiore è il rischio che il percorso vada a sovrapporsi ad un possibile ostacolo.

Risulta perciò necessario affidarsi, in presenza di ostacoli, all'algoritmo di Dubins andando a selezionare un opportuno raggio di curvatura R_2 a seconda del robot e della tipologia di ostacoli presenti (forma e dimensione).

7.6 Controllo collisione tra robot ed ostacolo

In questa Sezione viene descritto in che modo si determina la possibile collisione tra robot ed ostacolo.

Come si può vedere dallo schema in Figura 7.10, mediante l'algoritmo di Dubins viene determinato il percorso che porta il robot dal punto P_A al punto P_B. Tale percorso non è altro che un elenco di punti nel piano (il primo punto dell'elenco sarà P_A e l'ultimo sarà P_B).

Anche l'area occupata dagli ostacoli viene definita come un elenco dei punti, in modo analogo a quanto viene fatto per i nodi dello spazio libero nella fase di Learning.

Quindi, per poter determinare se il percorso va ad intercettare un ostacolo, il primo passo consiste nel confrontare i due elenchi, quello dei punti che costituiscono il percorso e quello dei punti che definiscono gli ostacoli: se uno o più punti del percorso corrispondono a uno o più punti degli ostacoli allora vuol dire che il percorso analizzato porta all'urto tra robot ed ostacolo. In tal caso sarà necessario andare a calcolare un nuovo percorso che vada dal punto P_A al punto $P_B - 1$.

Non basta però controllare la coincidenza dei punti dei due elenchi per stimare il possibile urto. Da una parte infatti va tenuto in considerazione l'ingombro del robot per cui i punti occupati dal robot non sono solo quelli della curva indicante il percorso da seguire, come già indicato nella Sezione 7.1. Dall'altra va tenuto in considerazione che gli ostacoli non sono puntiformi, ma anch'essi occupano una certa superficie sul piano (come si vede nelle Figure 7.11 e 7.12). Inoltre, in questi casi è opportuno accrescere virtualmente l'ingombro occupato dagli ostacoli per ragioni di sicurezza, in modo da ridurre il rischio di collisioni dovute ad una stima sbagliata dell'ostacolo o della geometria del robot.

Occorre perciò introdurre un margine che, nell'analisi del possibile urto robot-ostacolo, tenga in considerazione i due aspetti indicati: da una parte il fatto che il robot è caratterizzato da un certo ingombro, dall'altra che gli ostacoli possono essere più grandi del singolo punto indicato sulla mappa (o della griglia di punti indicata sulla mappa). Tale margine corrisponde a:

$$marg = s + s_{OBS}$$

s: ingombro del robot (vedere Sezione 7.1);
$s_{OBS} = 0.5 \cdot 1.25\ (m)$: superficie occupata dall'ostacolo;

Quindi il percorso va ad intersecare un ostacolo quando:

$$x_{OBS} - marg \leq x_P \leq x_{OBS} + marg \quad \& \quad y_{OBS} - marg \leq y_P \leq y_{OBS} + marg$$

Dove (x_P, y_P) sono le coordinate del punto del percorso, mentre (x_{OBS}, y_{OBS}) sono le coordinate di un punto che costituisce il centro dell'ostacolo.

8. Capitolo 8: Inseguimento di traiettoria e risultati ottenuti

Analizzando i risultati prodotti dal sistema di controllo (Capitolo 3) si può notare come l'azione di tali controlli produca uno scostamento tra il percorso di riferimento ed il percorso realizzato dal modello cinematico del robot. Inoltre, non è detto che il robot parta esattamente in corrispondenza del punto iniziale del percorso (P_o).

Per queste ragioni risulta indispensabile introdurre un inseguitore di traiettoria, anche noto come path tracker.

8.1 Pure Pursuit

Il Pure Pursuit (inseguimento puro) [43, 44] è un algoritmo di inseguimento del percorso che si basa sul calcolo della curvatura che permette al veicolo si spostarsi dalla sua posizione attuale ad una posizione obiettivo in modo tale da ricondurlo sul percorso voluto. La curvatura così ottenuta può essere utilizzata per calcolare i comandi di velocità lineare e angolare da dare al robot. Il principio di questo l'algoritmo è quello di scegliere una posizione obiettivo (punto G) sul percorso situata ad una certa distanza davanti al veicolo. Durante l'avanzamento del veicolo l'algoritmo sposta questo punto lungo il percorso in base alla posizione corrente del robot. Il parametro che indica la distanza a cui viene posizionato il punto di previsione viene indicato come *lookahead distance*. Il nome Pure Pursuit deriva dall'analogia che viene usata per descrivere questo metodo: guardando il risultato che si ottiene si ha l'impressione che il veicolo stia inseguendo un punto in movimento posizionato davanti a sé. Questa analogia viene spesso utilizzata per confrontare questo metodo con il modo in cui si guidano le automobili: durante la guida il guidatore guarda un punto immaginario che si trova ad una certa distanza davanti all'auto e si dirige verso quel punto per seguire la carreggiata.

Nella Figura 8.1 viene rappresentato il modello del robot mobile articolato ed il percorso che questo robot deve raggiungere muovendosi in retromarcia.

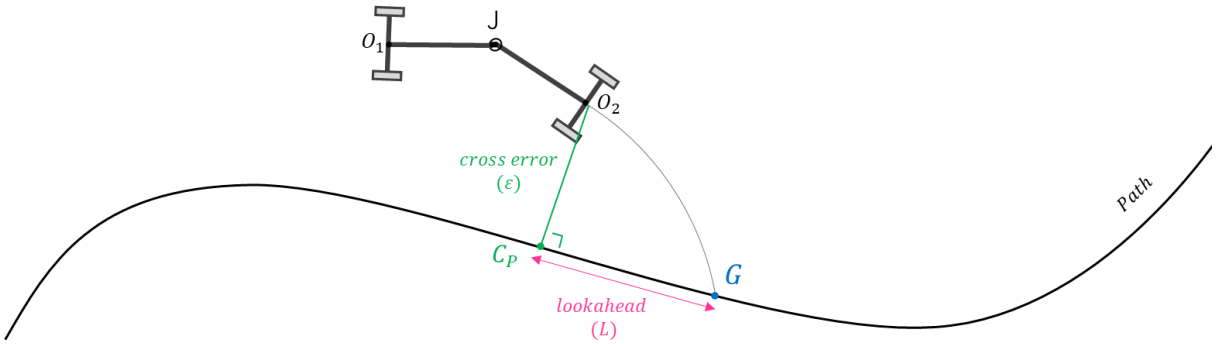

Figura 8.1 – Schematizzazione del robot mobile che insegue il percorso muovendosi all'indietro

Per comprendere il funzionamento dell'algoritmo è importante analizzare il sistema di coordinate di riferimento. Il percorso è memorizzato come un insieme di punti discreti (waypoints) definiti rispetto al sistema di riferimento globale x_0, y_0. I waypoints $(P_1, P_2, \dots, P_n)$ sono quindi indicati tramite coordinate (x, y).

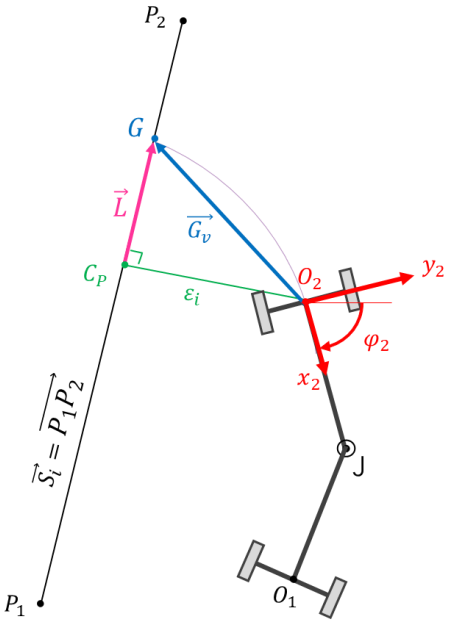

Figura 8.2 – Sistema di riferimento del modulo posteriore (in rosso) e grandezze di riferimento per l'inseguimento

L'approccio del Pure Pursuit consiste quindi nel determinare geometricamente la curvatura che guiderà il veicolo verso un punto del percorso prescelto, chiamato punto G, che si trova ad una distanza di lookahead $(\vec{L})$ dalla proiezione della posizione corrente del veicolo sul percorso (Figura 8.2).

L'implementazione dell'algoritmo è abbastanza semplice e può essere schematizzata come segue [44,45]:

1) Definizione della posizione e dell'orientazione attuale del veicolo in coordinate globali

La posa del veicolo è un elemento di input per l'algoritmo di inseguimento; poiché si ragiona in retromarcia si prende come riferimento la posa del modulo posteriore (punto O_2) del robot, espressa come (x_2, y_2, φ_2). φ_2 indica l'orientamento angolare del robot misurato in senso antiorario in radianti rispetto dall'asse x_0 (nella Figura 8.2 l'angolo φ_2 è negativo).

2) Determinazione del punto del percorso più vicino al veicolo

Il veicolo deve sterzare verso il punto obiettivo G più vicino. Pertanto, per prima cosa bisogna trovare il punto C_P, situato sul percorso, che sia più vicino al veicolo. C_P è il punto che si ottiene proiettando il punto O_2 sul segmento $\overrightarrow{S_i}$.

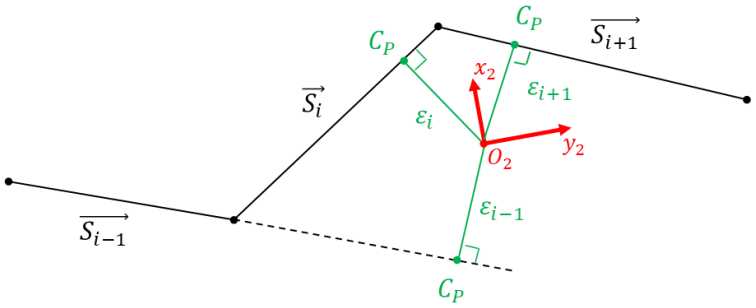

Figura 8.3 – Schema del sistema di riferimento del rimorchio e dei segmenti elementari che costituiscono il percorso

È perciò necessario trovare il segmento $\overrightarrow{S_i}$ (porzione elementare del percorso) che presenti il tratto ε_i più corto. Dove ε_i è la distanza tra i punti O_2 e C_P (Figura 8.3).

Per determinare il punto del percorso più vicino al robot si prendono in considerazione un certo numero[4] di segmenti $\overrightarrow{S_i}$ consecutivi e, per ognuno di essi, si vanno a calcolare le seguenti grandezze, riportate in Figura 8.4.

[4] Si potrebbero considerare tutti i segmenti $\overrightarrow{S_i}$ che costituiscono il percorso, ma per velocizzare il processo si fa l'assunzione euristica di controllarne solo una parte.

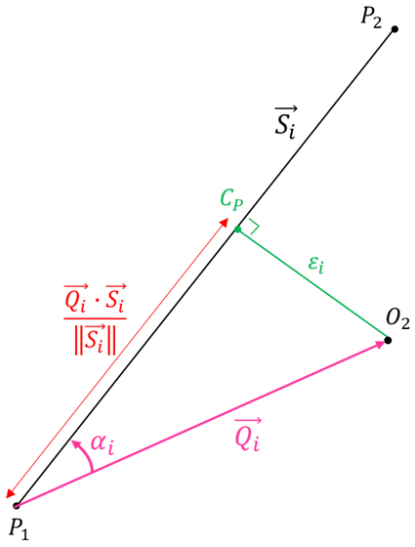

Figura 8.4 – Schema delle grandezze del segmento i-esimo del percorso

Si definisce $\overrightarrow{S_i}$ come il vettore che va dal waypoint P_1 del percorso al waypoint successivo P_2.

$$\overrightarrow{S_i} = \overrightarrow{P_1P_2} = \begin{bmatrix} x_S \\ y_S \end{bmatrix}$$

Si definisce inoltre il vettore $\overrightarrow{Q_i}$ che unisce il punto P_1 al punto di riferimento del veicolo (O_2).

$$\overrightarrow{Q_i} = \overrightarrow{P_1O_2} = \begin{bmatrix} x_Q \\ y_Q \end{bmatrix}$$

Per calcolare l'angolo α_i tra i vettori $\overrightarrow{S_i}$ e $\overrightarrow{Q_i}$ si ricorre all'espressione:

$$\alpha_i = \tan^{-1}\left(\frac{x_Q y_S - y_Q x_S}{x_Q x_S + y_Q y_S}\right)$$

Risulta ora possibile calcolare il segmento ε_i come:

$$\varepsilon_i = -\|\overrightarrow{Q_i}\| \sin \alpha_i$$

Il segno "–" è posto in modo tale da definire ε_i come un vettore che parte da O_2 e va verso il segmento $\overrightarrow{S_i}$.

$\|\overrightarrow{Q_i}\|$ rappresenta la norma del vettore $\|\overrightarrow{Q_i}\| = \sqrt{{x_Q}^2 + {y_Q}^2}$.

ε_i viene spesso definito come *cross error,* ossia l'errore laterale del veicolo rispetto al percorso.

Il punto C_P è calcolabile come:

$$C_P = P_1 + \frac{\overrightarrow{Q_i} \cdot \overrightarrow{S_i}}{\|\overrightarrow{S_i}\|} \widehat{S_i}$$

Dove $\frac{\vec{Q_i}\cdot\vec{S_i}}{\|\vec{S_i}\|}\hat{S_i}$ rappresenta il modulo della proiezione del vettore $\vec{Q_i}$ sul versore $\hat{S_i}$. Il versore viene calcolato come $\hat{S_i} = \frac{\vec{S_i}}{\|\vec{S_i}\|}$. Quindi il punto C_P si ottiene partendo dal punto P_1 e spostandosi lungo la direzione di $\vec{S_i}$ di una lunghezza pari alla proiezione del vettore $\vec{Q_i}$ su $\vec{S_i}$.

Dopo aver trovato le varie distanze ε_i dei segmenti $\vec{S_i}$ queste vengono confrontate tra loro. Il segmento $\vec{S_i}$ che presenta la distanza ε_i più corta rappresenta il tratto di percorso più vicino al veicolo; perciò, su questo si andrà a ricercare il punto obiettivo G.

3) <u>Determinare il punto obiettivo G in coordinate globali</u>

Il punto obiettivo si trova partendo da C_P e spostandosi lungo il percorso di una quantità pari alla lookahead distance (vettore $\vec{L}$), come mostrato nella Figura 8.5. Poiché i punti del percorso sono definiti rispetto al sistema di riferimento globale, il calcolo per determinare G viene eseguito in coordinate globali.

$$G = P_1 + \left(\frac{\vec{Q_i}\cdot\vec{S_i}}{\|\vec{S_i}\|} + L\right)\hat{S_i}$$

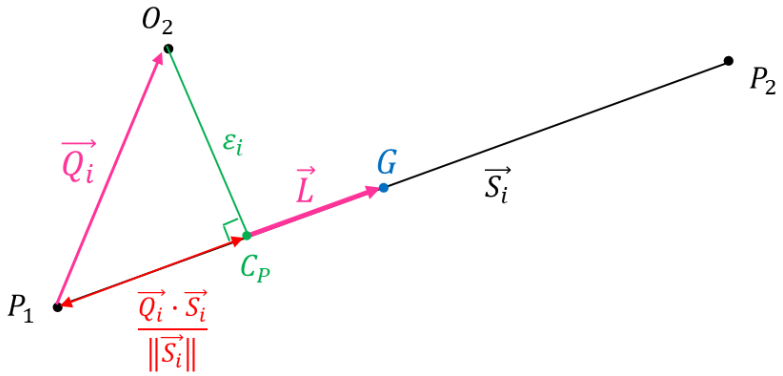

Figura 8.5 – Schema per il calcolo del punto obiettivo G

Bisogna però porre l'attenzione su due casi particolari, in cui il segmento rilevante, cioè il segmento $\vec{S_i}$ sul quale viene calcolato il punto G, deve essere scelto con cura:

$$Caso\ 1$$

$$se\ \frac{\overrightarrow{Q_i}\cdot\overrightarrow{S_i}}{\|\overrightarrow{S_i}\|}<0 \quad \Rightarrow \quad \overrightarrow{S_i}\ \text{è}\ rilevante$$

$$G=\begin{cases} P_1+\left(\frac{\overrightarrow{Q_i}\cdot\overrightarrow{S_i}}{\|\overrightarrow{S_i}\|}+L\right)\widehat{S_i} & se\ L>\frac{\overrightarrow{Q_i}\cdot\overrightarrow{S_i}}{\|\overrightarrow{S_i}\|} \\ P_1 & se\ L<\frac{\overrightarrow{Q_i}\cdot\overrightarrow{S_i}}{\|\overrightarrow{S_i}\|} \end{cases}$$

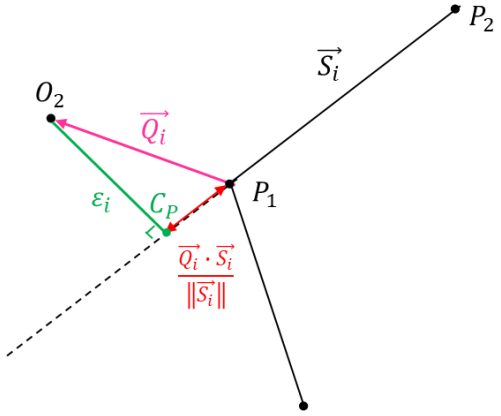

Figura 8.6 – Caso 1

$$Caso\ 2$$

$$se\ \frac{\overrightarrow{Q_i}\cdot\overrightarrow{S_i}}{\|\overrightarrow{S_i}\|}+L>\|\overrightarrow{S_i}\|\Rightarrow\overrightarrow{S_{i+1}}\ \text{è}\ rilevante$$

$$G=P_1+\left(\frac{\overrightarrow{Q_{i+1}}\cdot\overrightarrow{S_{i+1}}}{\|\overrightarrow{S_{i+1}}\|}+L\right)\widehat{S_{i+1}}$$

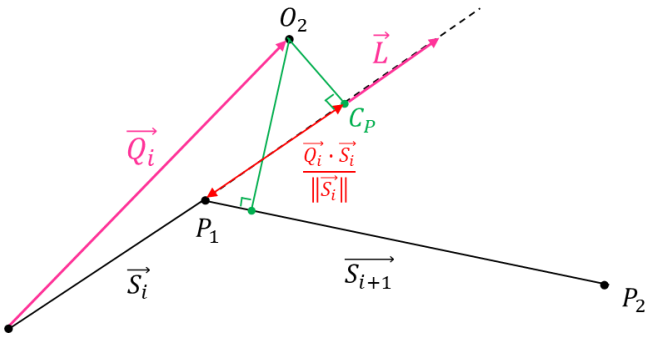

Figura 8.7 – Caso 2

4) <u>Trasformazione delle coordinate del punto obiettivo in coordinate del veicolo</u>

Una volta trovato il punto G, deve essere trasformato nelle coordinate locali del veicolo poiché per il calcolo della curvatura è necessario conoscere le coordinate x_G, y_G del vettore $\overrightarrow{G_v}$.

$$G(x_{G,0}, y_{G,0}) \ \Rightarrow \ \overrightarrow{G_v}(x_G, y_G)$$

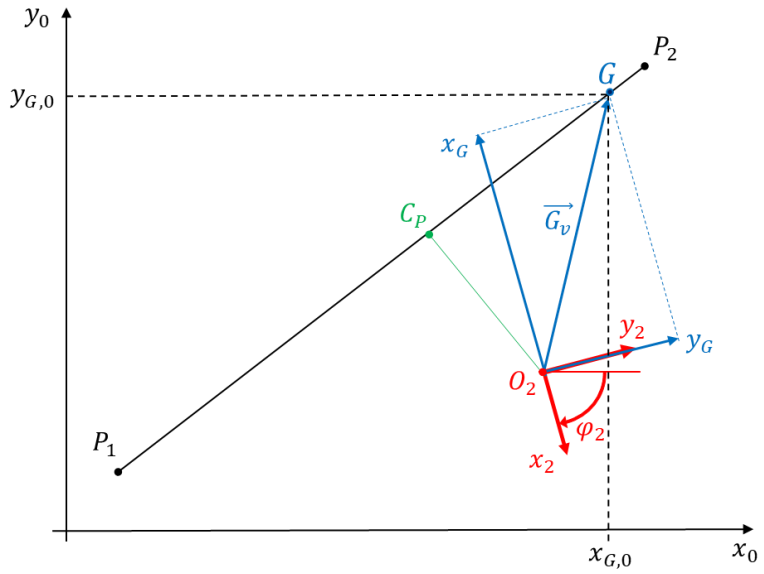

Figura 8.8 – Schema per il calcolo del vettore $\overrightarrow{G_v}$

Per ottenere il cambio di coordinate si può fare riferimento alla Figura 8.8 ottenendo:

$$\overrightarrow{G_v} = \begin{bmatrix} x_G \\ y_G \end{bmatrix} = \begin{bmatrix} -\cos\varphi_2 & -\sin\varphi_2 \\ -\sin\varphi_2 & \cos\varphi_2 \end{bmatrix} \cdot \begin{bmatrix} x_{G,0} - x_2 \\ y_{G,0} - y_2 \end{bmatrix}$$

5) <u>Calcolare la curvatura</u>

Analizzando la Figura 8.9 è possibile definire la curvatura del veicolo desiderata per raggiungere il percorso.

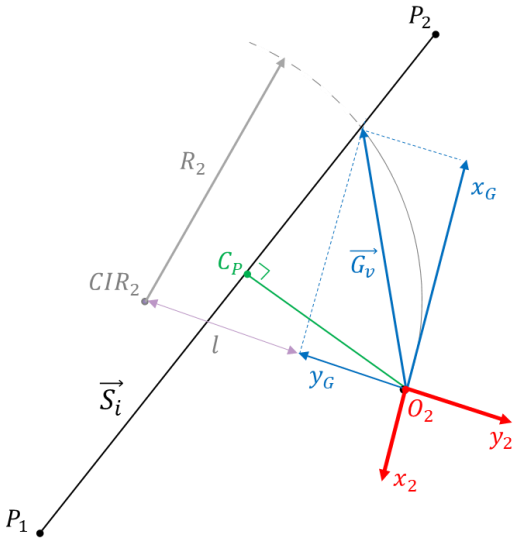

Figura 8.9 – Schema per il calcolo del raggio di curvatura R_2

$$\begin{cases} R_2^2 = l^2 + x_G^2 \\ R_2 = l + y_G \end{cases} \quad \Rightarrow \quad 0 = x_G^2 + y_G^2 - 2R_2\, y_G$$

Poiché

$$x_G^2 + y_G^2 = \left\|\overrightarrow{G_v}\right\|^2$$

È possibile esprimere il raggio di curvatura R_2 e la corrispondente curvatura ρ_2 come:

$$R_2 = \frac{\left\|\overrightarrow{G_v}\right\|^2}{2y_G} \qquad \rho_2 = \frac{2y_G}{\left\|\overrightarrow{G_v}\right\|^2}$$

Si può osservare come il segno della curvatura (se il robot gira in senso orario o antiorario) dipenda dal segno assunto da y_G.

La curvatura, riferita al modulo posteriore, viene quindi trasformata dal controller di bordo del veicolo nei riferimenti di velocità angolare delle due ruote dell'avantreno (o nei riferimenti di velocità longitudinale ed angolare) dai blocchi descritti nel Capitolo 5. In questo modo si ottiene il comando in termini di velocità che sposta il robot dalla sua posizione attuale per raggiungere un punto G in modo da seguire il percorso.

8.1.1 Effetti della modifica della lookahead distance

La lookahead distance indica la distanza di previsione per definire il punto obiettivo G sul percorso che il robot dovrebbe seguire partendo dalla sua posizione corrente. È la proprietà di ottimizzazione principale per il controller, per questo la lookahead distance è un parametro importante all'interno nell'algoritmo Pure Pursuit. Gli effetti della modifica di questa distanza vanno considerati attentamente in quanto possono cambiare il modo in cui il robot segue il percorso. In quest'ottica emergono due obiettivi principali:

1) **Recuperare il percorso**: se il veicolo è ad una certa distanza dal percorso e deve raggiungerlo.

2) **Mantenere il percorso**: se il veicolo è sul percorso e vuole rimanere su di esso.

Per riguadagnare rapidamente il percorso tra i waypoint, una piccola lookahead distance permette di far muovere rapidamente il robot verso il percorso. Tuttavia, come si può vedere nella Figura 8.10 a sinistra, il robot supera il percorso e oscilla lungo esso. Invece distanze più lunghe tendono a convergere sul percorso in modo più graduale e con meno oscillazioni

(Figura 8.10 a destra), ma potrebbero portare ad avere raggi di curvatura maggiori vicino agli angoli.

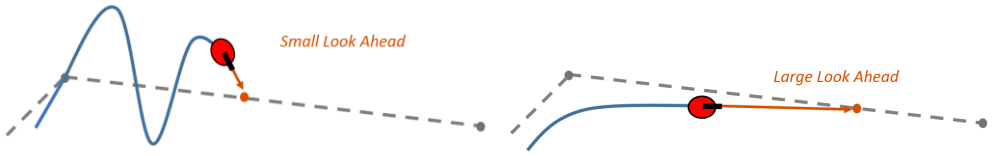

Figura 8.10 – Effetti di una piccola lookahead distance (a sinistra) e di una grande lookahead distance (a destra)

La lookahead distance deve perciò essere ottimizzata a seconda del tipo di applicazione e delle caratteristiche del robot. Nel caso dei robot Epi.q ed Agri.q sono stati adottati i seguenti valori:

Tabella 8.1 – Valori della lookahead distance a seconda del tipo di robot e dei raggi di curvatura del percorso

Robot	**Raggi di curvatura lungo il percorso**	**Lookahead Distance[5] [m]**
Epi.q	$R_2 < 1\ m$	$0.25 \cdot v_{max}$
Epi.q	$R_2 > 1\ m$	$0.35 \cdot v_{max}$
Agri.q	$R_2 < 4\ m$	$0.80 \cdot v_{max}$
Agri.q	$R_2 > 4\ m$	v_{max}

8.1.2 Limitazioni dell'inseguimento puro

L'algoritmo di puro inseguimento presenta alcune limitazioni:

- L'algoritmo non permette di seguire esattamente un percorso composto da una serie di spezzate; questo è dovuto principalmente al fatto che il modello cinematico (anche quello dinamico) non può seguire traiettorie discontinue in modo esatto quindi l'inseguitore tende a raccordare il percorso. La lookahead distance deve essere scelta pertanto in modo da ottimizzare le prestazioni e convergere sul percorso nel minor tempo possibile.
- L'algoritmo non stabilizza il robot in un punto. Per questo dove essere applicato un ulteriore controllo che, una volta raggiunto il punto finale P_R, comandi l'arresto del robot.

[5] La costante che moltiplica la velocità massima ha unità di misura in secondi. In questo caso la lookahead distance viene definita come la distanza percorsa dal robot che si muove alla massima velocità (v_{max}) in tot secondi.

- Vi sono dei limiti legati agli effetti della dinamica. Il metodo Pure Pursuit non modella la capacità del veicolo o dei suoi attuatori, e quindi presuppone una risposta perfetta alle curvature richieste. Questo causa due problemi in quanto nella realtà il veicolo non si chiuderà sulla traiettoria con la rapidità desiderata.

Nonostante siano presenti questi limiti l'algoritmo Pure Pursuit è tra i metodi più utilizzati: la tesi di laurea di Amidi [46] contiene i risultati del confronto tra diversi algoritmi di tracciamento. I test su questi algoritmi hanno mostrato che il metodo Pure Pursuit è tra i più robusti ed affidabili in uso.

8.2 Risultati ottenuti

Di seguito vengono analizzati 4 esempi (due per Epi.q e due per Agri.q) in cui vengono mostrati i risultati ottenuti sfruttando i sistemi di controllo del moto in retromarcia ed i pianificatori del percorso ottimale descritti nei capitoli precedenti. In tutti e 4 i casi viene anche richiesto di avere i due moduli allineati al termine del percorso in marcia indietro.

8.2.1 Esempio 1: Robot Epi.q, moto in retromarcia, percorso senza ostacoli

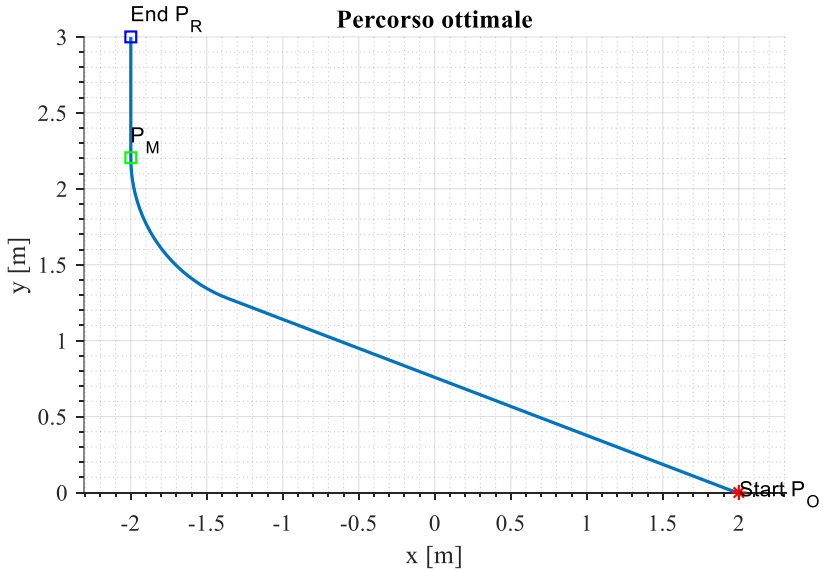

Figura 8.11 – Percorso di riferimento da far percorrere al robot Epi.q in retromarcia

Tabella 8.2 – Grandezze caratteristiche del percorso e della simulazione

Start $[x_{2,O},\ y_{2,O},\ \varphi_{2,O},\ \delta_O]$	$[2 \quad 0 \quad -10 \quad 3]$	$m\ \ m\ deg\ deg$
Goal $[x_{2,R},\ y_{2,R},\ \varphi_{2,R},\ \delta_R]$	$[-2 \quad 3 \quad -90 \quad 0]$	$m\ \ m\ deg\ deg$
Lunghezza del percorso	5.59	m
Tempo di percorrenza	19.7	s
Posizione finale modulo posteriore	$[-2.02 \quad 2.95 \quad -95 \quad -1.8]$	$m\ m\ deg\ deg$
Accelerazione centripeta massima	0.22	m/s^2

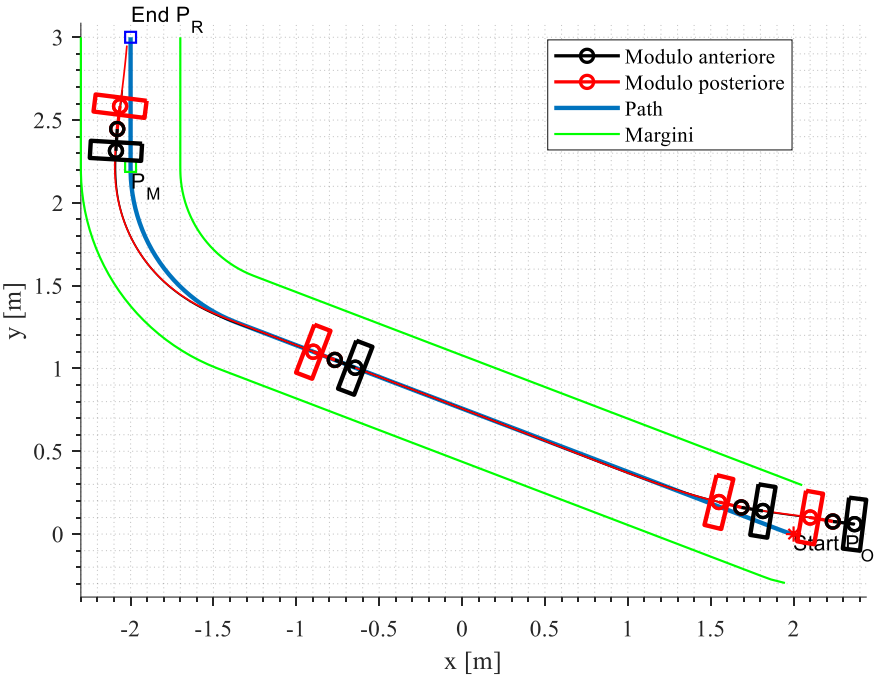

Figura 8.12 – Rappresentazione del moto in retromarcia di Epi.q

La Figura 8.12 mostra i movimenti dell'intero veicolo e le traiettorie disegnate dai punti fondamentali: O_1 per il modulo anteriore (in nero), O_2 per il rimorchio (in rosso).

Si può notare come il robot non parta in corrispondenza del punto P_O; questo permette di osservare chiaramente l'intervento dell'inseguitore di traiettoria, il quale provvede a riportare il robot sul percorso. Lo stesso si può notare in corrispondenza della curva finale: l'azione del controllo dell'angolo δ porta il robot a divergere dal percorso stabilito, ma grazie all'inseguitore di traiettoria la traiettoria viene recuperata.

I margini in verde servono per avere una indicazione di massima sullo spazio che può essere occupato dal robot senza divergere eccessivamente dalla traiettoria imposta (sono

maggiori dell'ingombro del robot s descritto nella Sezione 7.1). Nel caso del robot Epi.q la larghezza di questa banda corrisponde a 60 cm.

Osservando la Tabella 8.2 si può vedere come il robot raggiunga il punto finale con buona approssimazione sia in termini di posizione che di orientazione.

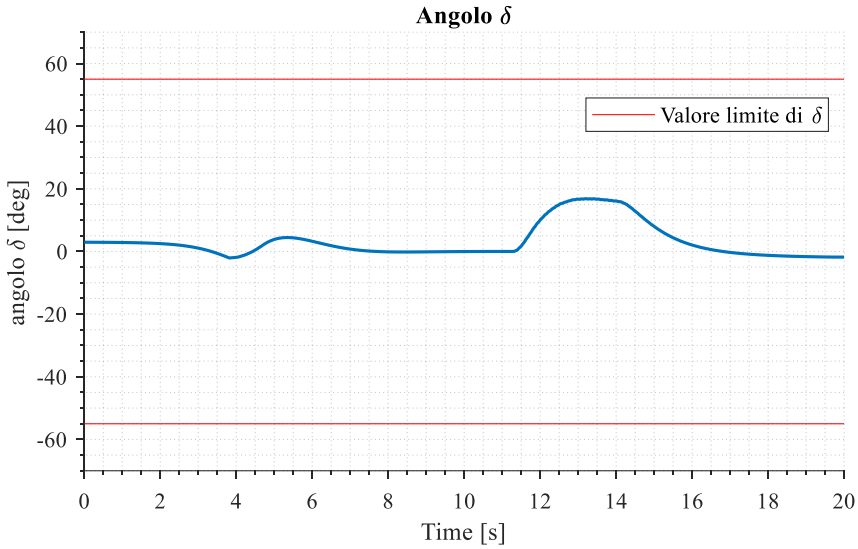

Figura 8.13 – Andamento dell'angolo δ nel tempo

L'andamento di δ nel tempo mostrato in Figura 8.13 ha lo scopo di evidenziare il rispetto del vincolo $|\delta| \leq \delta_{max}$ durante tutta la percorrenza della traiettoria; infatti, l'andamento di δ (in blu) è sempre compreso fra le due rette (in rosso) rappresentanti il vincolo $(-\delta_{max}, \delta_{max})$.

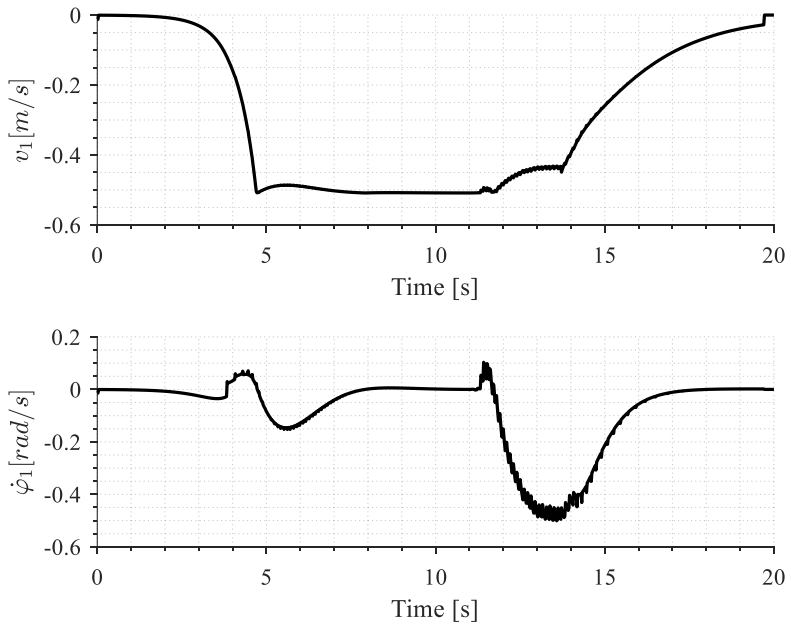

Figura 8.14 – Andamento della velocità longitudinale ed angolare del modulo anteriore nel tempo

Osservando la Figura 8.14 si possono notare delle oscillazioni nell'andamento della velocità angolare $\dot{\varphi}_1$. Questo aspetto è dovuto al fatto che il sistema di controllo agisce in modo da cercare di mantenere l'angolo δ attorno ad una condizione di equilibrio instabile.

La velocità longitudinale v_1 presenta un andamento oscillatorio meno evidente in quanto, per il robot Epi.q, questa velocità ha un contributo minore sul controllo dell'angolo δ.

Dall'andamento della velocità longitudinale v_1 si può vedere come sia stata impostata la velocità angolare ω (discussa nel Capitolo 5) in modo da ottenere una fase di accelerazione inziale ed una fase di decelerazione verso il termine del percorso. Questo aspetto è visibile anche dagli andamenti delle velocità angolari delle due ruote riportati nella Figura 8.15.

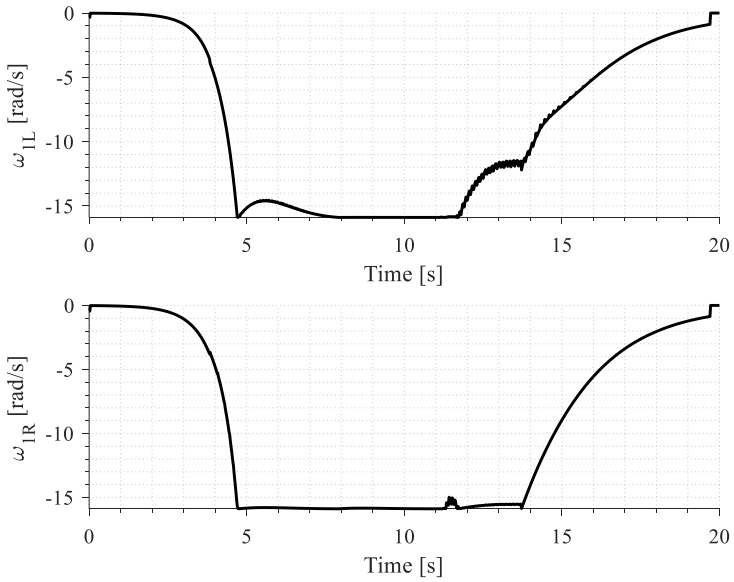

Figura 8.15 – Andamento delle velocità angolari delle ruote del modulo anteriore nel tempo

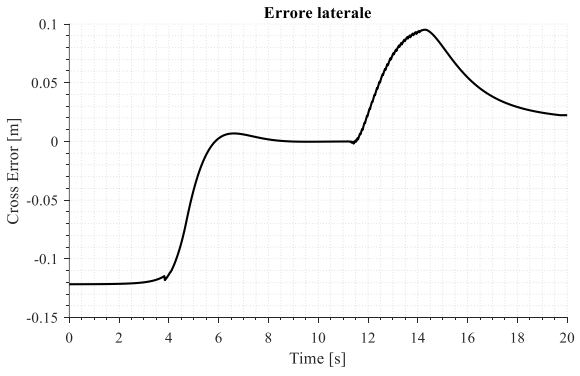

Figura 8.16 – Andamento dell'errore laterale nel tempo

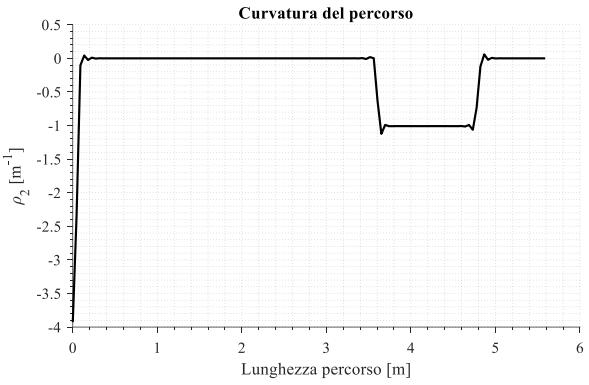

Figura 8.17 – Andamento della curvatura ρ_2 del percorso lungo la lunghezza di quest'ultimo

Analizzando l'andamento dell'errore laterale (Figura 8.16), l'andamento della curvatura (Figura 8.17) e la Figura 8.12, si può vedere che, esclusa la fase inziale in cui il robot si porta sul percorso, si ha un aumento dell'errore laterale in corrispondenza della curva (indicata da un aumento in valore assoluto della curvatura ρ_2). Comunque il cross error rimane molto contenuto (inferiore a 15 cm) ed il robot non esce dai margini rappresentati nella Figura 8.12 in verde.

8.2.2 Esempio 2: Robot Agri.q, moto in retromarcia, percorso senza ostacoli

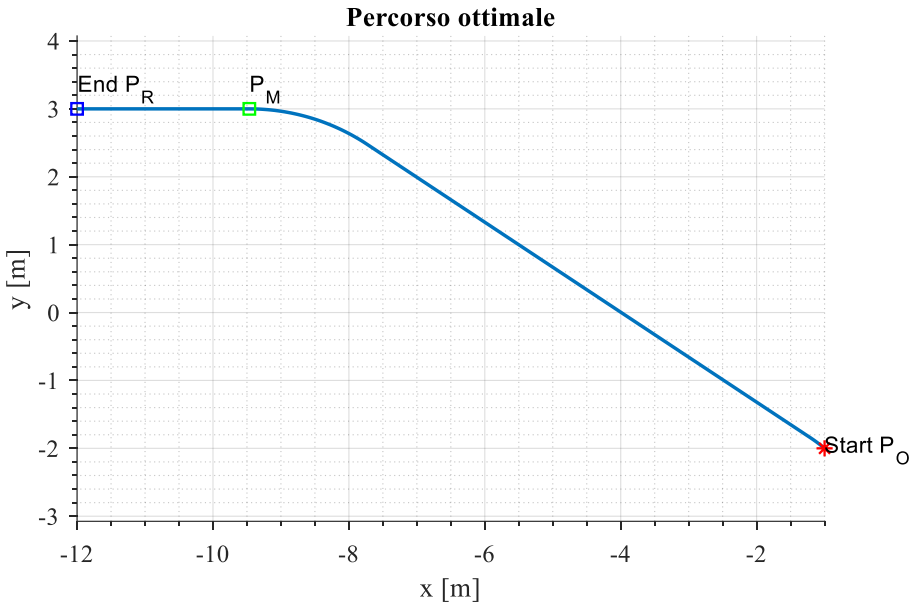

Figura 8.18 – Percorso di riferimento da far percorrere al robot Agri.q in retromarcia

Tabella 8.3 – Grandezze caratteristiche del percorso e della simulazione

Start $[x_{2,O},\ y_{2,O},\ \varphi_{2,O},\ \delta_O]$	$[-1 \quad -2 \quad -40 \quad 5]$	$m \ \ m \ deg \ deg$
Goal $[x_{2,R},\ y_{2,R},\ \varphi_{2,R},\ \delta_R]$	$[-12 \quad 3 \quad 0 \quad 0]$	$m \ \ m \ deg \ deg$
Lunghezza del percorso	12.5	m
Tempo di percorrenza	25.8	s
Posizione finale modulo posteriore	$[-11.9 \quad 3 \quad 1.5 \quad 2.2]$	$m \ m \ deg \ deg$
Accelerazione centripeta massima	0.14	m/s^2

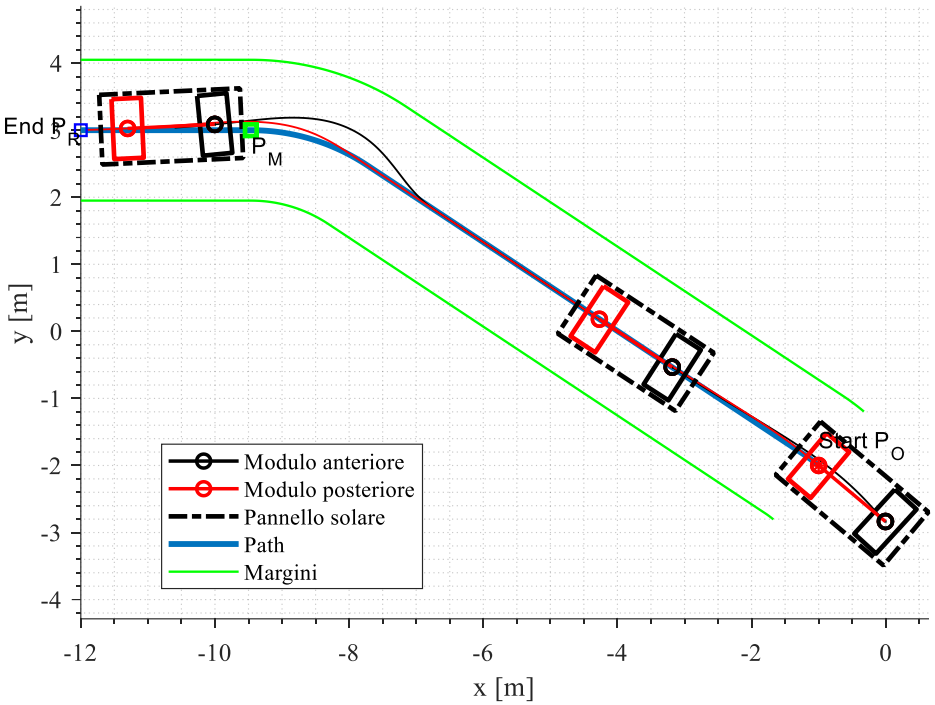

Figura 8.19 – Rappresentazione del moto in retromarcia di Agri.q

La Figura 8.19 mostra i movimenti del robot Agri.q e le traiettorie disegnate dai punti O_1 per il modulo anteriore (in nero), O_2 per il rimorchio (in rosso). In questo caso viene anche riportato l'ingombro del pannello solare posto sul rover; questa scelta permette di individuare in modo semplice lo spazio che il robot occupa sul piano in cui deve muoversi. Si controlla quindi che il pannello solare, indice dell'ingombro del robot, non superi i margini (in verde): la banda di sicurezza per il robot Agri.q è posta pari a 2.1 m.

In questo caso il robot parte in corrispondenza del punto iniziale del percorso, ma orientato in modo diverso; anche in questo caso l'intervento dell'inseguitore permette di correggere l'orientazione del robot in modo da indirizzarlo correttamente sul percorso. Lo stesso si può notare in corrispondenza della curva finale: l'azione del controllo dell'angolo δ porta il robot a divergere leggermente dal percorso stabilito (la traiettoria del modulo posteriore, in rosso, non è più in corrispondenza del percorso, in blu), ma grazie all'inseguitore Pure Pursuit la traiettoria viene recuperata.

Osservando la Tabella 8.3 si può vedere come il robot raggiunga il punto finale con buona approssimazione sia in termini di posizione che di orientazione.

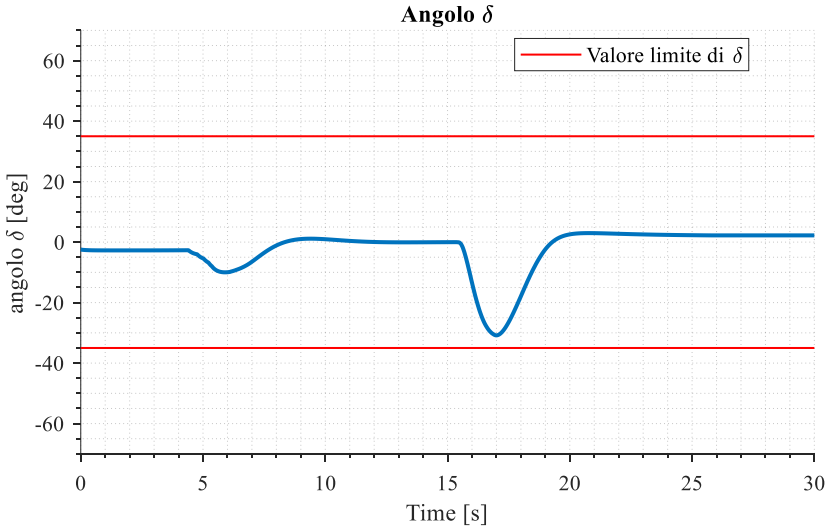

Figura 8.20 – Andamento dell'angolo δ nel tempo

L'andamento di δ nel tempo mostrato in Figura 8.20 evidenzia il rispetto del vincolo $|\delta| \leq \delta_{max}$ durante la percorrenza della traiettoria: l'andamento di δ (in blu) è sempre compreso fra le due rette (in rosso) rappresentanti il vincolo $(-\delta_{max}, \delta_{max})$. Ciò è reso possibile dal controllo su δ che ha il compito di stabilizzare tale angolo attorno alla soluzione δ_{eq} instabile. Si può apprezzare inoltre come le potenzialità di movimento siano sfruttate appieno, il che è riscontrabile dall'approssimarsi dell'andamento di δ in alcuni punti al limite massimo δ_{max}.

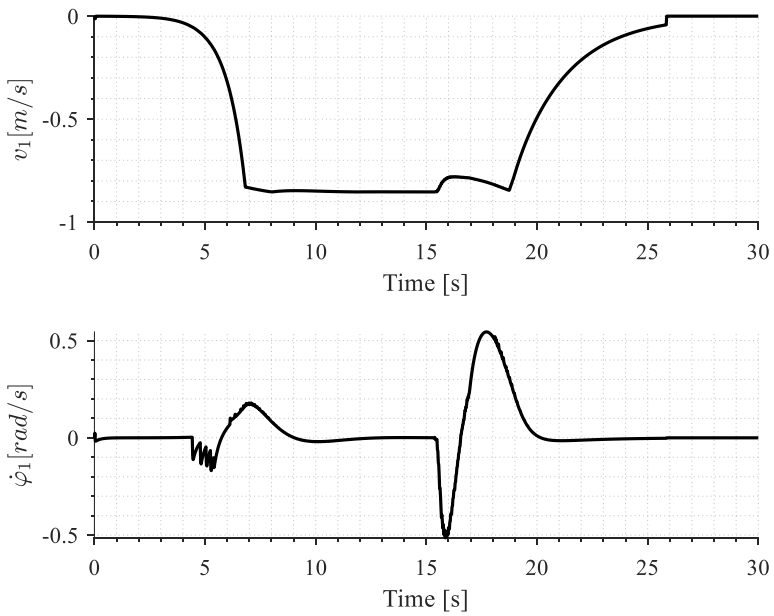

Figura 8.21 – Andamento della velocità longitudinale ed angolare del modulo anteriore nel tempo

Osservando la Figura 8.21 si possono notare delle oscillazioni nell'andamento della velocità angolare $\dot{\varphi}_1$. Questo aspetto è dovuto al fatto che il sistema di controllo dell'angolo δ agisce in modo da cercare di mantenere tale angolo attorno ad una condizione di equilibrio instabile. Nel caso di Agri.q l'andamento oscillatorio di $\dot{\varphi}_1$ è più contenuto perché, essendo presente il controllo diretto sull'angolo δ, si può intervenire sia sulla parte proporzionale che derivativa di quest'ultimo in modo da attenuare le oscillazioni.

L'andamento della velocità longitudinale invece non presenta un andamento oscillatorio perché il controllore non interviene su questa velocità.

Per completezza vengono anche riportati gli andamenti delle velocità angolari delle ruote del modulo anteriore (Figura 8.22).

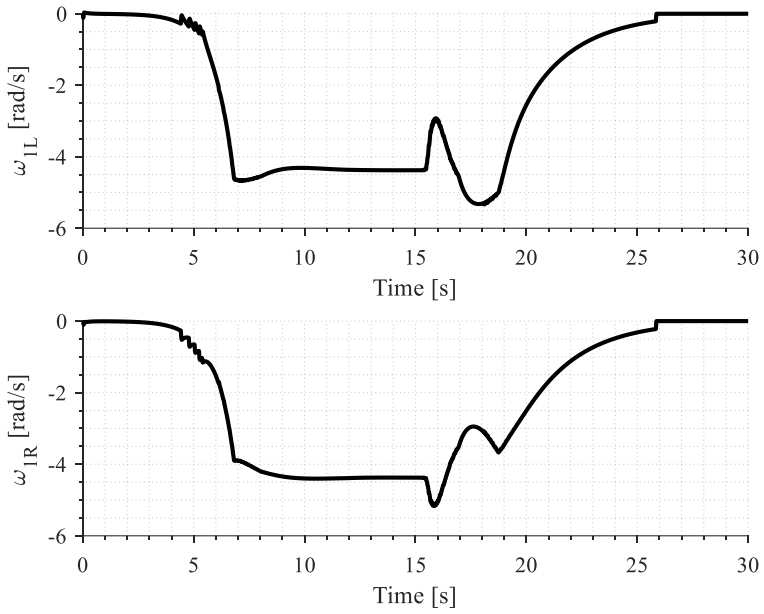

Figura 8.22 – Andamento delle velocità angolari delle ruote del modulo anteriore nel tempo

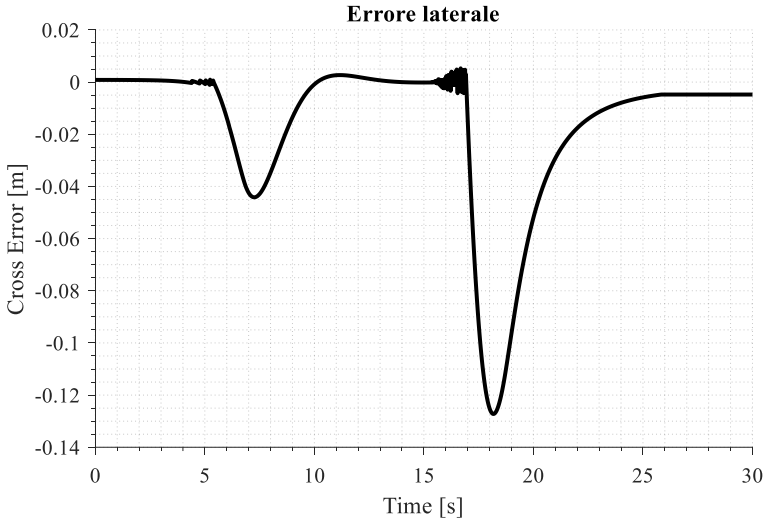

Figura 8.23 – Andamento dell'errore laterale nel tempo

In questo esempio il massimo errore laterale si raggiunge in corrisponde della curva presente al termine del percorso (Figura 8.23). Anche in questo caso il cross error rimane molto contenuto, segno che il percorso ottimale e la definizione della lookahead distance dell'inseguitore di traiettoria sono stati definiti in modo corretto.

Si può osservare inoltre un andamento oscillatorio dell'errore laterale al tempo $t = 5\ s$ e al tempo $t = 17s$: poiché si sta controllando un sistema instabile è normale avere delle oscillazioni attorno al valore desiderato a meno di un controllo molto aggressivo che però potrebbe portare, nel caso reale, alla saturazione dei motori impedendo un controllo complessivo corretto.

8.2.3 Esempio 3: Robot Epi.q, moto in retromarcia, percorso con ostacoli

Tabella 8.4 – Grandezze caratteristiche del percorso e della simulazione

Fase di Learning	0.6	s
Fase di Quering	0.04	s
Start $[x_{2,0}, y_{2,0}, \varphi_{2,0}, \delta_0]$	$[1.5 \quad 2 \quad -90 \quad 0]$	$m \ m \ deg \ deg$
Goal $[x_{2,R}, y_{2,R}, \varphi_{2,R}, \delta_R]$	$[9.5 \quad 3 \quad 90 \quad 0]$	$m \ m \ deg \ deg$
Lunghezza del percorso	10.7	m
Tempo di percorrenza	30.2	s
Posizione finale modulo posteriore	$[9.5 \quad 3.05 \quad 85.13 \quad -0.57]$	$m \ m \ deg \ deg$
Accelerazione centripeta massima	0.24	m/s^2

Nota la posizione degli ostacoli, del punto iniziale e del punto finale, mediante l'algoritmo di pianificazione PRM è possibile determinare la successione di punti che permette di collegare lo Start al Goal evitando gli ostacoli, come visibile in Figura 8.24.

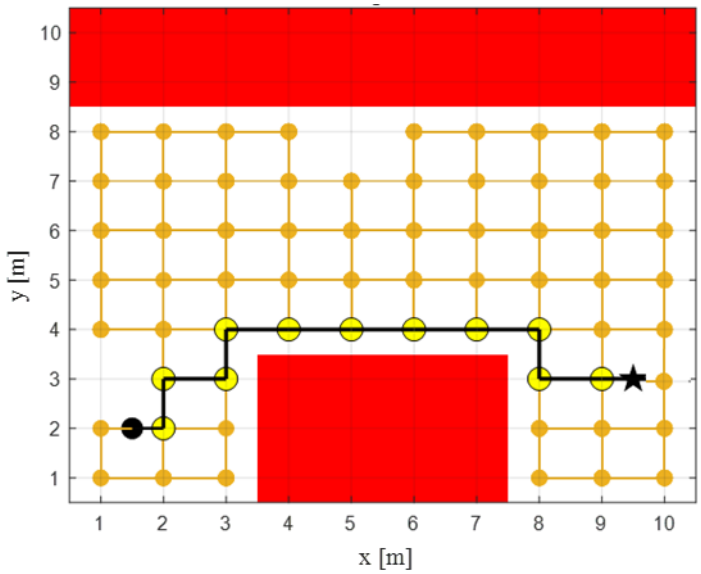

Figura 8.24 – Applicazione dell'algoritmo PRM per andare da Start (cerchio nero) a Goal (stella) evitando gli ostacoli (in rosso)

Una volta definita la successione di punti viene applicato l'algoritmo di interpolazione, descritto nella Sezione 7.5, in modo da determinare il percorso che permetta di raggiungere la configurazione desiderata senza urtare gli ostacoli ed evitando allo stesso tempo la generazione di curve inutili. Il percorso così ottenuto viene mostrato nella Figura 8.25.

Analizzando tale Figura è possibile osservare come il percorso (in blu) venga selezionato in modo da evitare la collisione robot-ostacolo, in quanto l'ingombro del robot (rappresentato dalle linee viola) non interseca gli ostacoli (rappresentati dai rettangoli rossi). Gli ostacoli inoltre vengono leggermente maggiorati, come si può notare confrontando le Figure 8.24 e 8.25, per ridurre ulteriormente il rischio di urto nel passaggio tra simulazione ed applicazione reale.

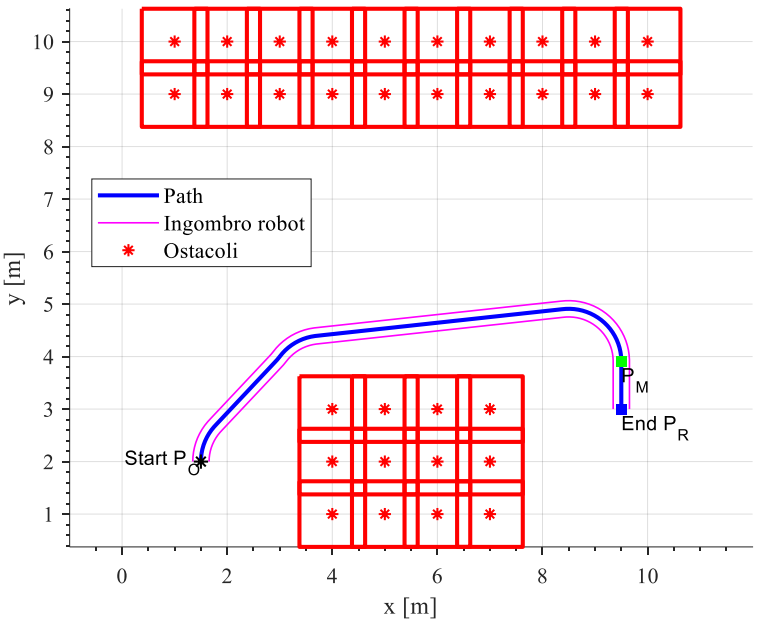

Figura 8.25 – Percorso ottenuto mediante l'algoritmo di interpolazione

Dopo aver trovato il percorso, che il robot dovrà eseguire in retromarcia, è possibile eseguire la simulazione in modo da verificare che il controllore gestista correttamente il moto all'indietro del robot Epi.q.

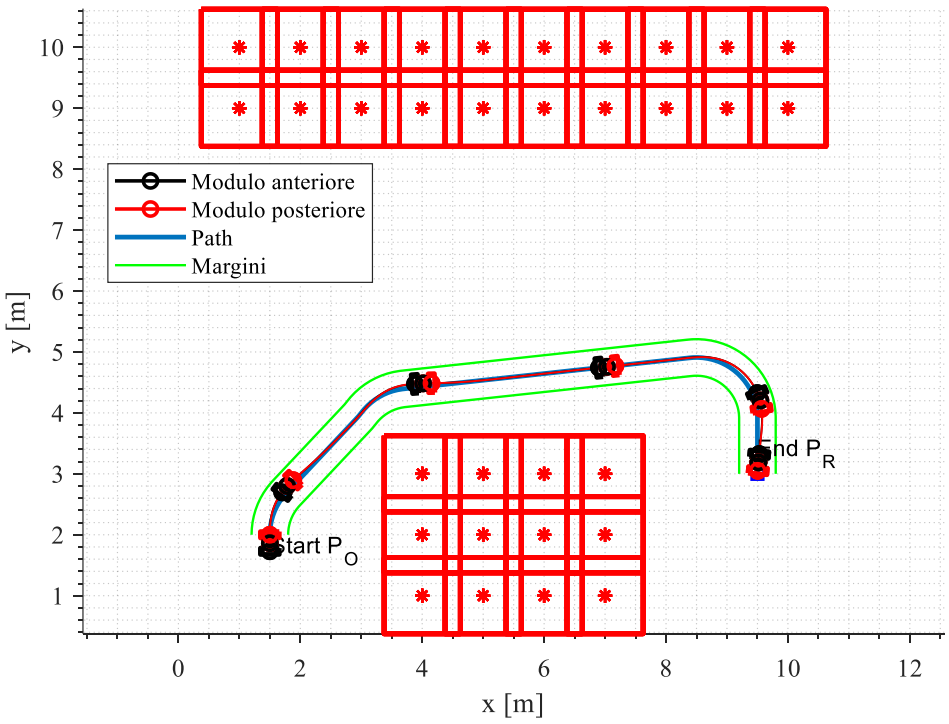

Figura 8.26 – Rappresentazione del moto in retromarcia di Epi.q

La Figura 8.26 mostra il moto in retromarcia di Epi.q e le traiettorie disegnate dai punti O_1 per il modulo anteriore (in nero), O_2 per il modulo posteriore (in rosso). I margini in verde differiscono da quelli viola presenti nella Figura 8.25: la fascia verde è maggiore di quella viola e serve per avere una indicazione sullo spazio che può essere occupato dal robot senza divergere eccessivamente dalla traiettoria imposta. Poiché il robot Epi.q, durante tutto il moto, è contenuto nella fascia verde, è possibile affermare che il robot Epi.q segue fedelmente il percorso assegnato. Ciò è possibile grazie ad un corretto bilanciamento tra il controllo del moto in retromarcia e l'inseguimento di traiettoria, visibile ancora più chiaramente nell'ingrandimento di Figura 8.27.

Ovviamente, per una maggiore sicurezza, durante il controllo dell'urto robot-ostacolo, nelle fasi di definizione del percorso, si potrebbero prendere in considerazione le dimensioni della banda verde (e non solo l'ingombro del robot) in modo da tenere in conto anche una possibile divergenza del robot dal percorso indicato. Si potrebbe anche fare un'analisi estesa per determinare quale sia il massimo valore di errore laterale e usarlo per ridefinire la larghezza dei margini.

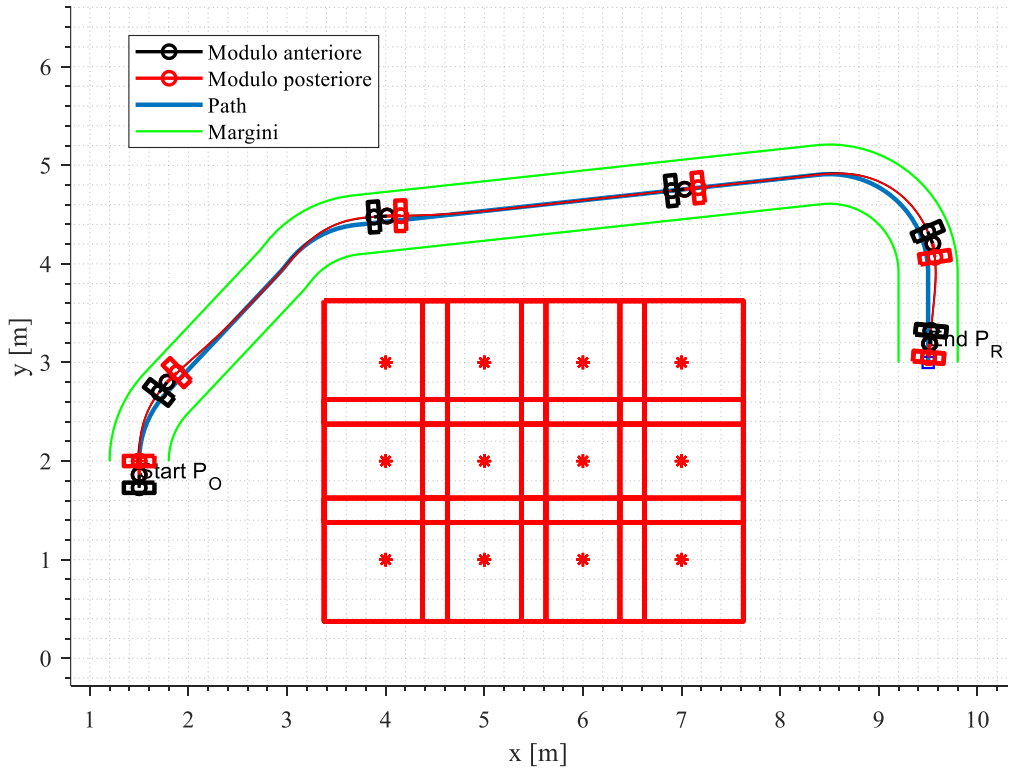

Figura 8.27 – Ingrandimento del moto in retromarcia di Epi.q

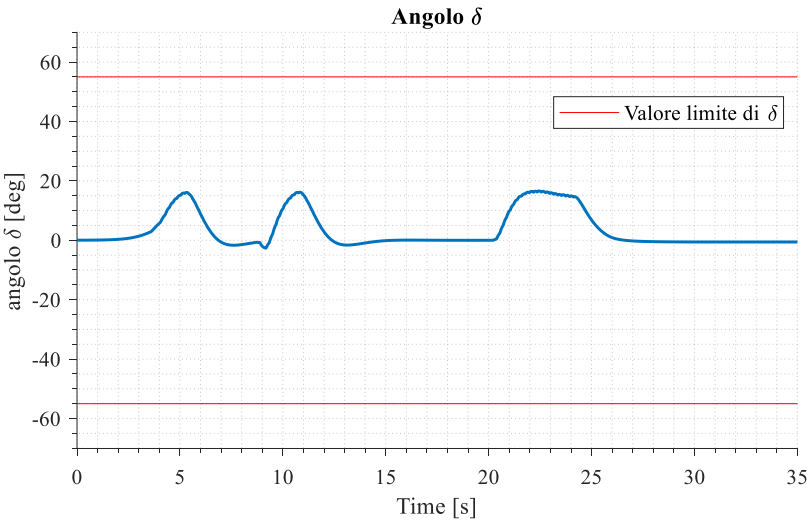

Figura 8.28 – Andamento dell'angolo δ nel tempo

L'andamento di δ nel tempo mostrato in Figura 8.28 evidenzia il rispetto del vincolo $|\delta| \leq \delta_{max}$ durante tutta la percorrenza della traiettoria.

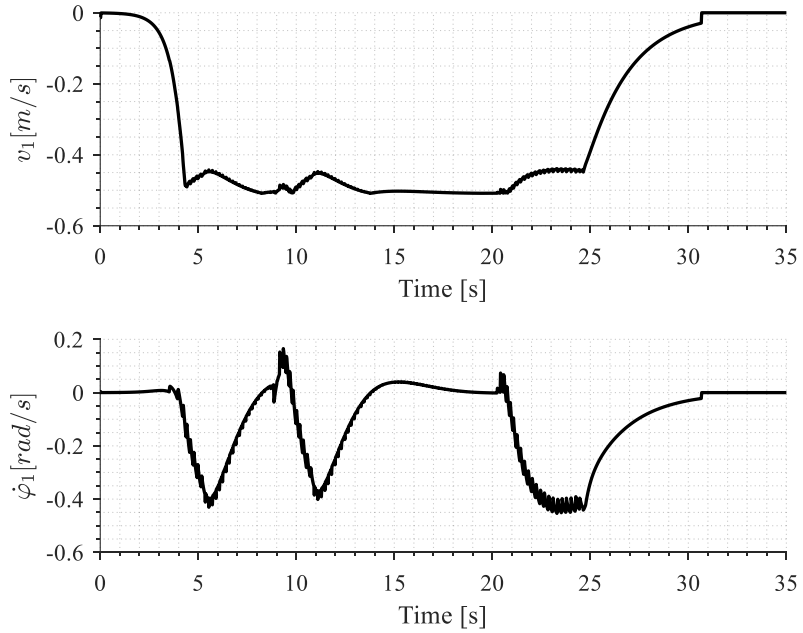

Figura 8.29 – Andamento della velocità longitudinale ed angolare del modulo anteriore nel tempo

Anche in questo caso il controllo del moto in retromarcia genera delle oscillazioni nell'andamento della velocità angolare $\dot{\varphi}_1$ (Figura 8.29).

Per completezza vengono anche riportati gli andamenti delle velocità angolari delle ruote del modulo anteriore (Figura 8.30).

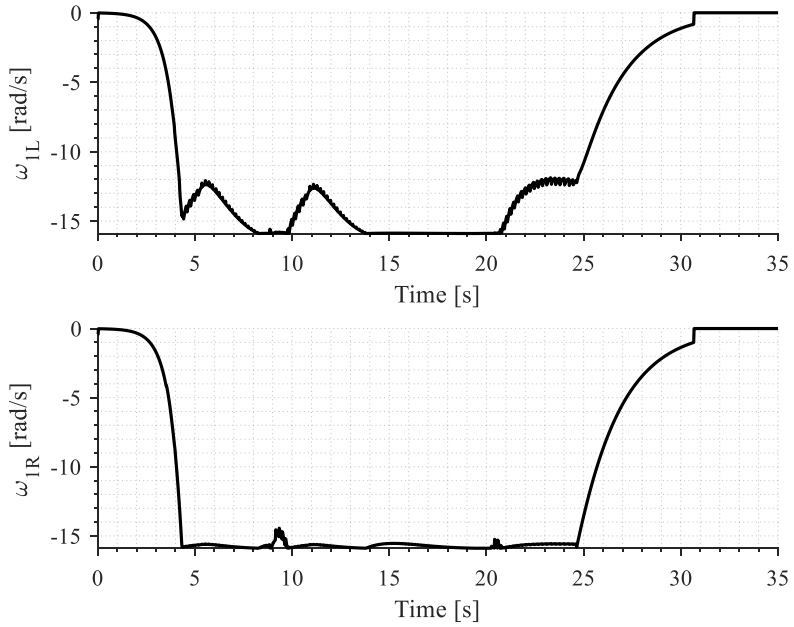

Figura 8.30 – Andamento delle velocità angolari delle ruote del modulo anteriore nel tempo

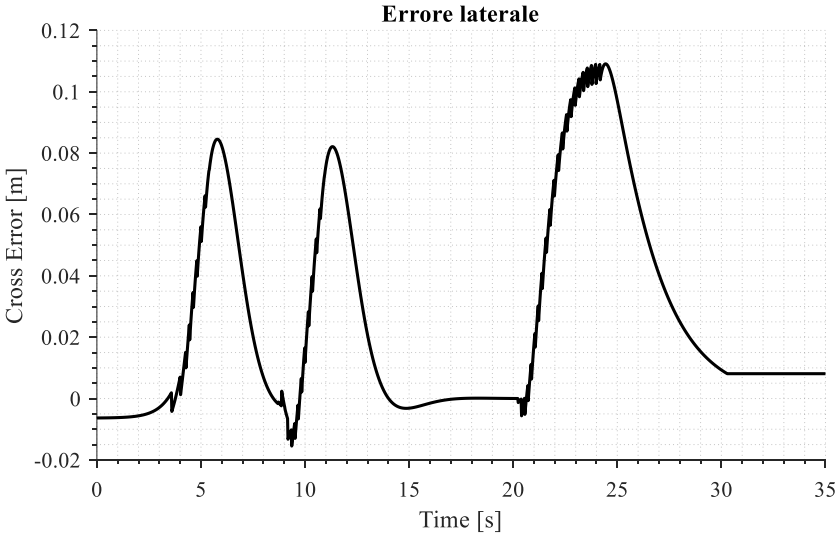

Figura 8.31 – Andamento dell'errore laterale nel tempo

L'andamento dell'errore laterale nel tempo mostrato in Figura 8.31 ha lo scopo di evidenziare il ridotto scostamento tra il robot ed il percorso, il quale non supera i 12 cm. Come nei casi precedenti l'errore laterale aumenta in corrispondenza delle curve.

8.2.4 Esempio 4: Robot Agri.q, moto in retromarcia, percorso con ostacoli

Tabella 8.5 – Grandezze caratteristiche del percorso e della simulazione

Fase di Learning	3.4	s
Fase di Quering	0.26	s
Start $[x_{2,0}, y_{2,0}, \varphi_{2,0}, \delta_0]$	$[6 \quad 2 \quad -90 \quad 0]$	$m \; m \; deg \; deg$
Goal $[x_{2,R}, y_{2,R}, \varphi_{2,R}, \delta_R]$	$[17.5 \quad 4 \quad 90 \quad 0]$	$m \; m \; deg \; deg$
Lunghezza del percorso	31.94	m
Tempo di percorrenza	52.4	s
Posizione finale modulo posteriore	$[17.52 \quad 4.10 \quad 86.8 \quad -3.94]$	$m \; m \; deg \; deg$
Accelerazione centripeta massima	0.35	m/s^2

Nota la posizione degli ostacoli, del punto iniziale e del punto finale, mediante l'algoritmo di pianificazione PRM è possibile determinare la successione di punti che permette di collegare lo Start al Goal evitando gli ostacoli, come visibile in Figura 8.32. in questa Figura si può vedere come non sempre l'algoritmo PRM trovi la successione di punti più breve. Questo aspetto viene successivamente corretto mediante l'interpolazione dei punti (Figura 8.33).

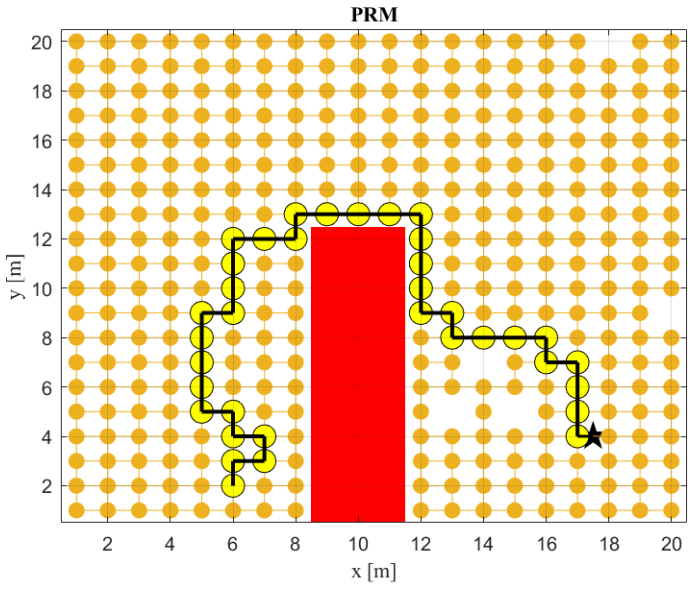

Figura 8.32 – Applicazione dell'algoritmo PRM per andare da Start (cerchio nero) a Goal (stella) evitando gli ostacoli (in rosso).

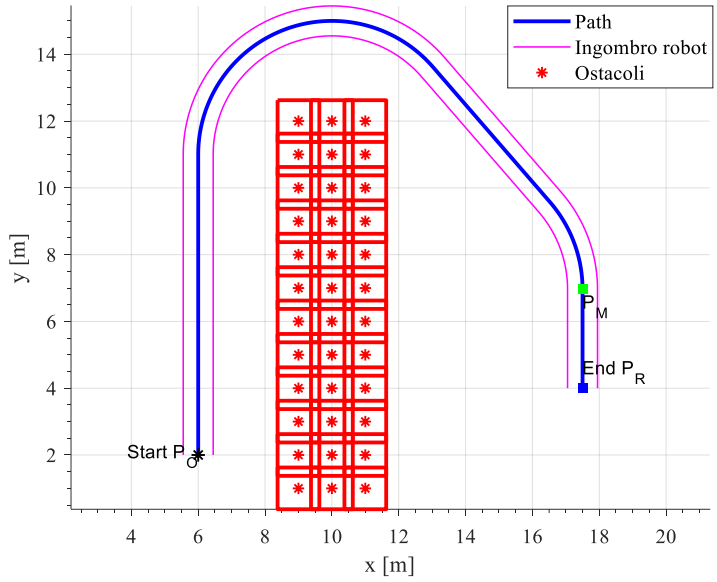

Figura 8.33 – Percorso ottenuto mediante l'algoritmo di interpolazione

Applicato l'algoritmo di interpolazione si determina il percorso per raggiungere la configurazione finale senza urtare gli ostacoli ed evitando allo stesso tempo la generazione di curve inutili. Il percorso così ottenuto viene mostrato nella Figura 8.33. Il percorso (in blu) viene definito in modo da evitare la collisione robot-ostacolo, in quanto l'ingombro del robot (linee viola) non interseca gli ostacoli (rettangoli rossi). Anche in questo caso la dimensione degli ostacoli è stata aumentata.

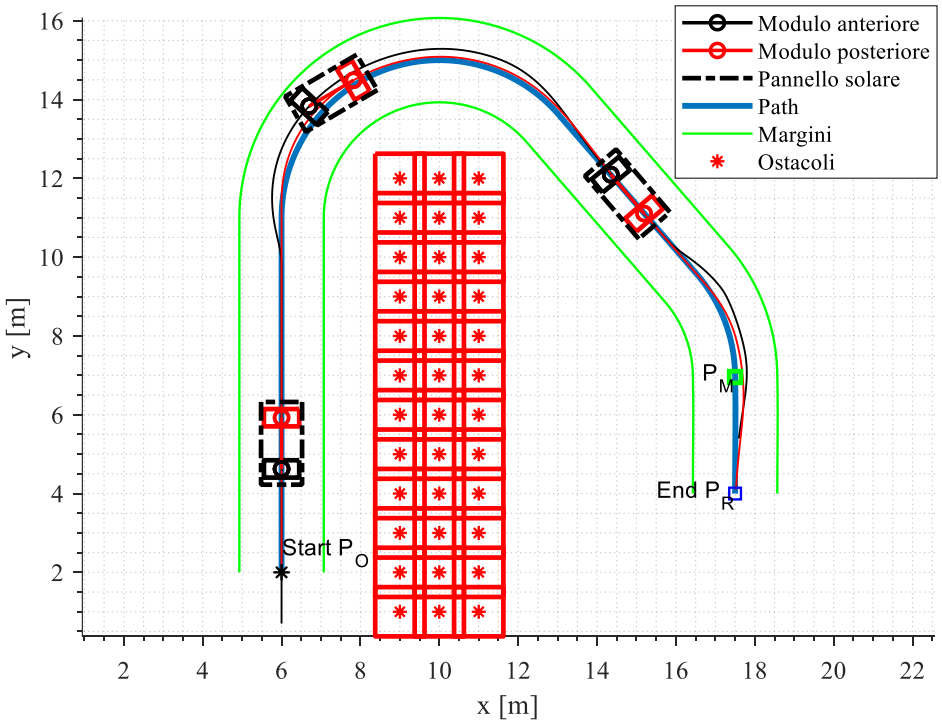

Figura 8.34 – Rappresentazione del moto in retromarcia di Agri.q

La Figura 8.34 mostra il moto in retromarcia di Agri.q e le traiettorie disegnate dai punti O_1 per il modulo anteriore (in nero), O_2 per il modulo posteriore (in rosso). Inoltre, viene riportata anche la sagoma del pannello solare, il quale viene preso come riferimento per capire se l'ingombro del robot resta compreso tra i margini di riferimento (linee verdi). Anche in questo caso la fascia verde è maggiore di quella viola e serve per avere una indicazione sullo spazio che può essere occupato dal robot senza divergere eccessivamente dalla traiettoria imposta. Poiché il pannello solare di Agri.q è contenuto tra i margini in verde, è possibile affermare che vi è un bilanciamento corretto tra i parametri del controllo dell'angolo δ (K_P) ed i parametri relativi all'inseguimento di traiettoria (distanza di lookahead L).

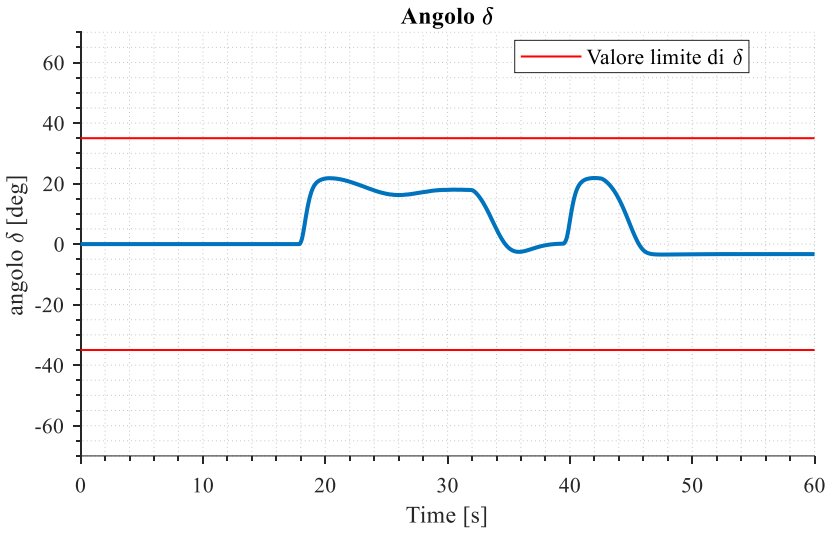

Figura 8.35 – Andamento dell'angolo δ nel tempo

L'andamento di δ nel tempo mostrato in Figura 8.35 evidenzia il rispetto del vincolo $|\delta| \leq \delta_{max}$ durante tutta la percorrenza della traiettoria.

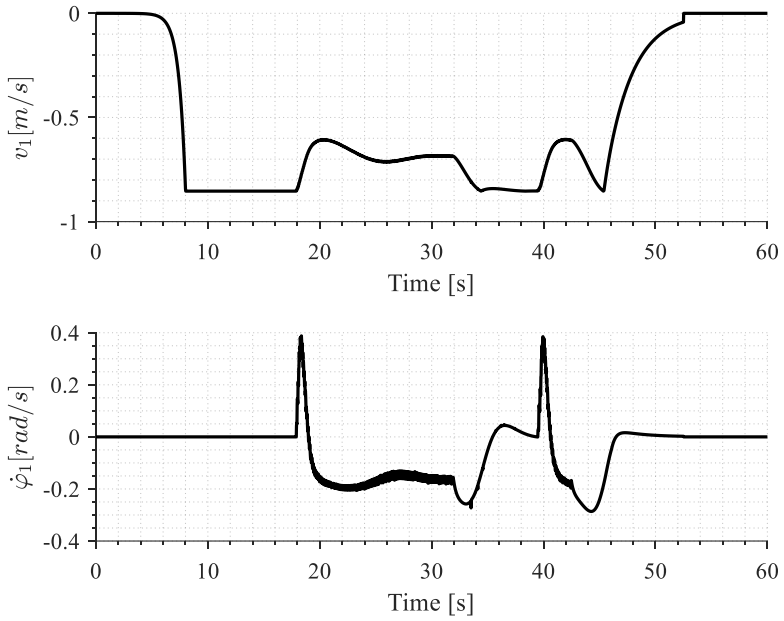

Figura 8.36 – Andamento della velocità longitudinale ed angolare del modulo anteriore nel tempo

Osservando la Figura 8.36 si possono notare delle oscillazioni nell'andamento della velocità angolare $\dot{\varphi}_1$. Questo aspetto è dovuto al fatto che il sistema di controllo agisce in modo da cercare di mantenere l'angolo δ attorno ad una condizione di equilibrio instabile. Per completezza vengono anche riportati gli andamenti delle velocità angolari delle ruote del modulo anteriore (Figura 8.37).

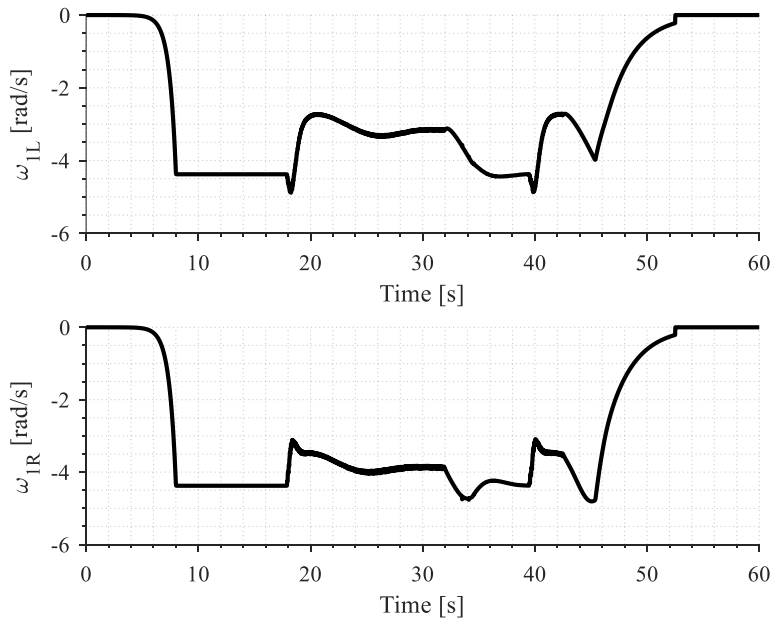

Figura 8.37 – Andamento delle velocità angolari delle ruote del modulo anteriore nel tempo

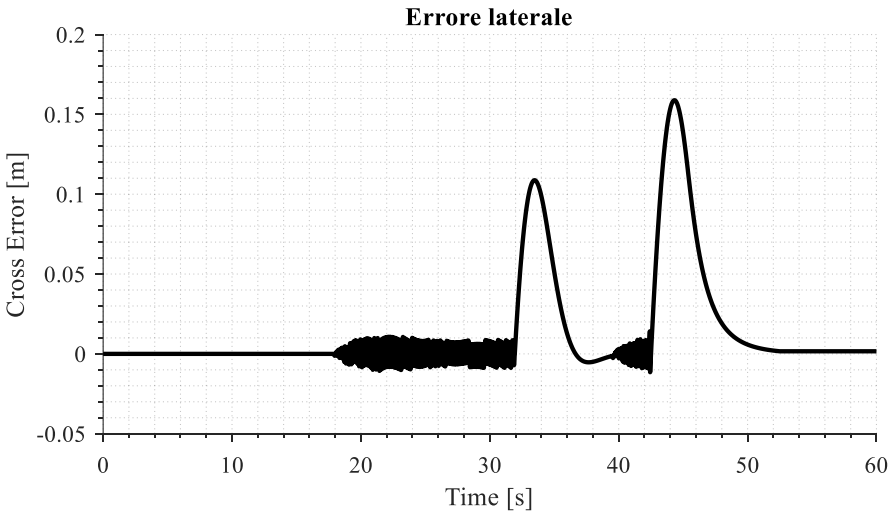

Figura 8.38 – Andamento dell'errore laterale nel tempo

L'andamento dell'errore laterale nel tempo, mostrato in Figura 8.38, mette in evidenza il ridotto scostamento tra il robot ed il percorso; inoltre mostra che, come nei casi precedenti, tale scostamento aumenta in corrispondenza delle curve.

9. Capitolo 9: Applicazione del controllo al modello dinamico del robot Agri.q

Dopo aver verificato il corretto funzionamento del controllo del moto in retromarcia sui modelli cinematici dei due robot, il passo successivo è attuare la verifica considerando i modelli dinamici. Questa verifica si rende indispensabile in quanto il modello cinematico non prende in considerazione diverse caratteristiche del robot reale. Gli aspetti più importanti, trascurati nel modello cinematico e che invece vengono implementati nel modello dinamico, sono:

- il modello di contatto ruota terreno e lo slittamento laterale delle ruote;
- la trasmissione anteriore bilaterale[6], scleronoma[7], non liscia[8];
- la trasmissione posteriore unilaterale, scleronoma, non liscia;
- la saturazione della coppia erogata dai motori.

In questo caso passare dal modello cinematico a quello dinamico corrisponde al passaggio da un modello bidimensionale (visto dall'alto) del robot ad un modello tridimensionale, realizzato mediante programma Adams (Figure 9.1, 9.2 e 9.3). Infatti, per verificare la validità del controllo, è stato utilizzato un modello dinamico del robot Agri.q realizzato in Adams. Fruttando la co-simulazione tra i programmi Adams e Simulink è possibile visionare il comportamento dinamico del robot nelle fasi di retromarcia sotto l'azione del controllore.

[6] La trasmissione bilaterale è un sistema che permette di trasferire coppia in entrambi i sensi di rotazione. La trasmissione unilaterale invece permette il trasferimento di coppia in una sola direzione di marcia.

[7] Il termine scleronomo indica un sistema i cui vincoli posizionali sono indipendenti dal tempo.

[8] Secondo la meccanica razionale un vincolo è definito liscio se la reazione che ne risulta è in ogni istante di tempo ortogonale allo spazio delle configurazioni. Una definizione più intuitiva è quella di definire i vincoli non dissipativi, o vincoli lisci, tutti i vincoli le cui reazioni vincolari compiono un lavoro virtuale non negativo, per ogni spostamento virtuale del sistema.

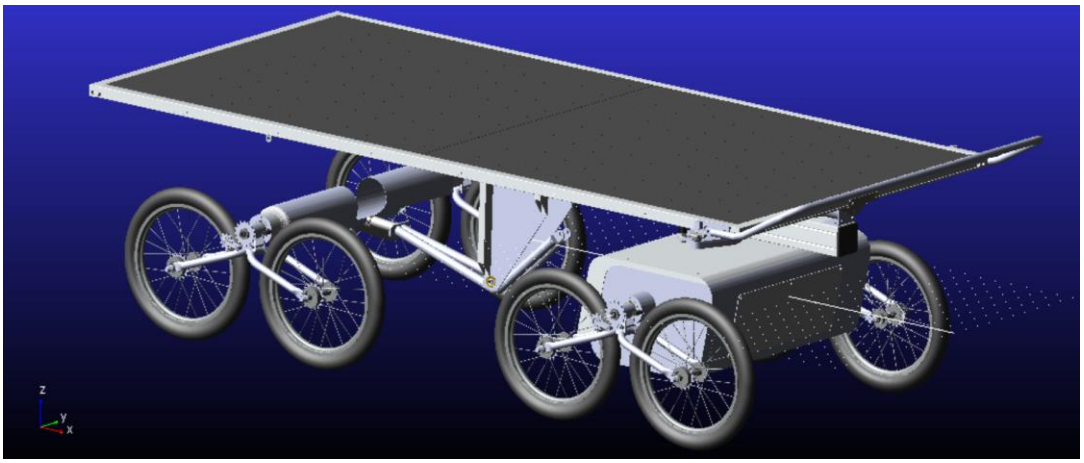

Figura 9.1 – Modello del robot Agri.q realizzato in ambiente Adams

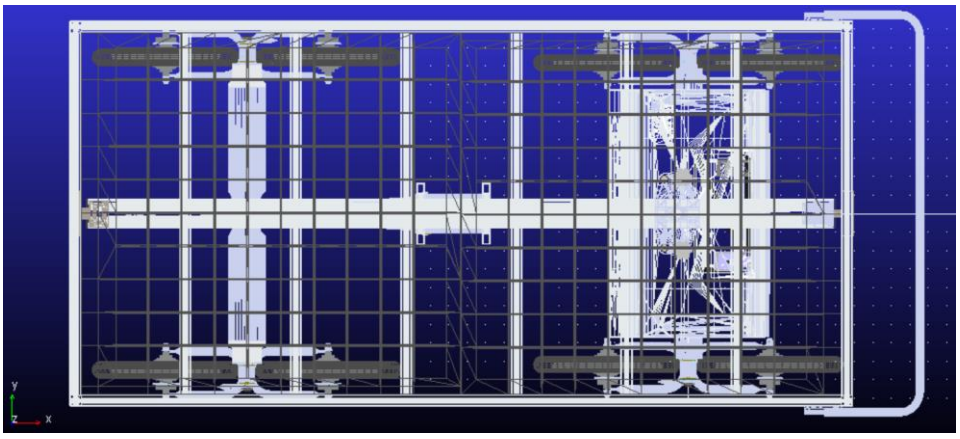

Figura 9.2 – Vista dall'alto del modello di Agri.q, il tipo di visualizzazione permette di vedere i moduli sotto il pannello solare

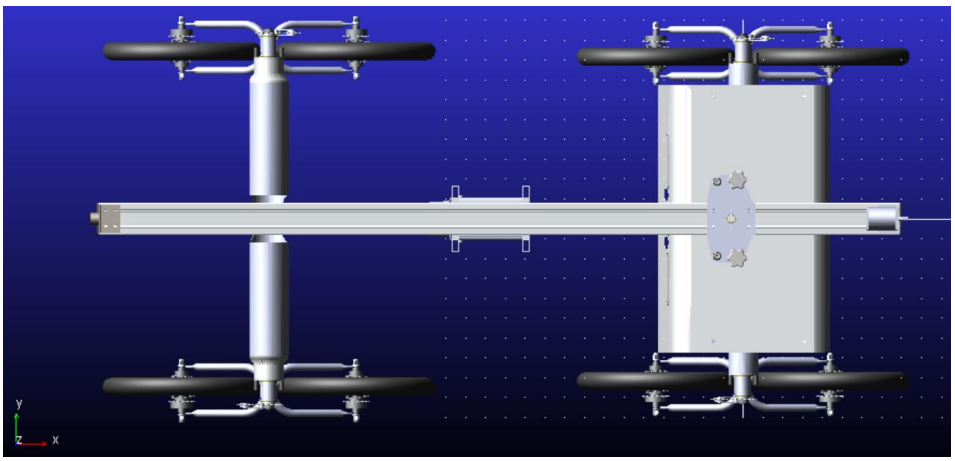

Figura 9.3 – Vista dall'alto del modello di Agri.q, il pannello solare viene rimosso in modo da vedere i due moduli sottostanti

Nei paragrafi successivi viene analizzato il modello dinamico del robot (valido sia per Epi.q che per Agri.q) e viene fornita una descrizione dei punti elencati (i quali fanno riferimento specificatamente al robot Agri.q)

9.1 Modello dinamico del robot

Per determinare le equazioni dinamiche che governano i due robot articolati viene ripreso lo studio [22], nel quale viene utilizzato l'approccio di Newton-Eulero. Il sistema è suddiviso in 10 sottosistemi (modulo anteriore, modulo posteriore e 8 ruote a contatto col suolo) e per ognuno di essi viene disegnato un diagramma a corpo libero (Figura 9.5). Questo vuol dire che le unità di locomozione non sono più semplificate considerando una ruota singola equivalente per ogni lato, ma vengono considerate tutte le ruote a contatto col terreno (Figura 9.4).

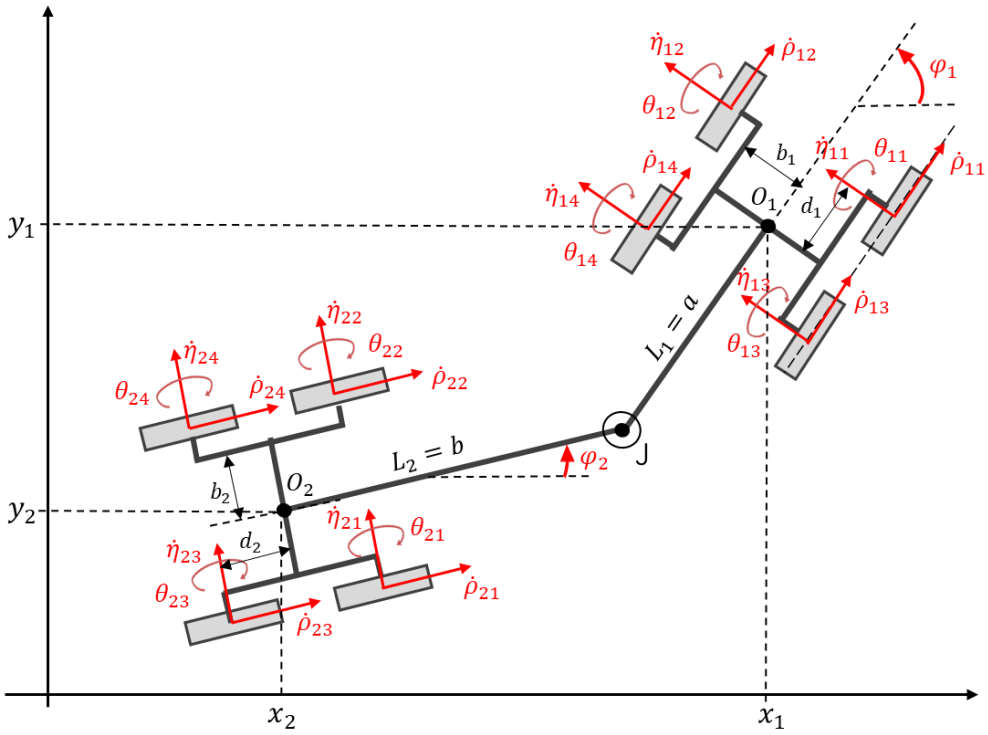

Figura 9.4 – Modello del robot mobile articolato con 8 ruote a contatto col terreno

Tabella 9.1 – Descrizione dei tipi di spostamento delle ruote

Simbolo	Descrizione
η_{jk}	Spostamento laterale della ruota jk nel sistema di riferimento locale
ρ_{jk}	Spostamento longitudinale della ruota jk nel sistema di riferimento locale
θ_{jk}	Spostamento angolare della ruota jk attorno al suo asse

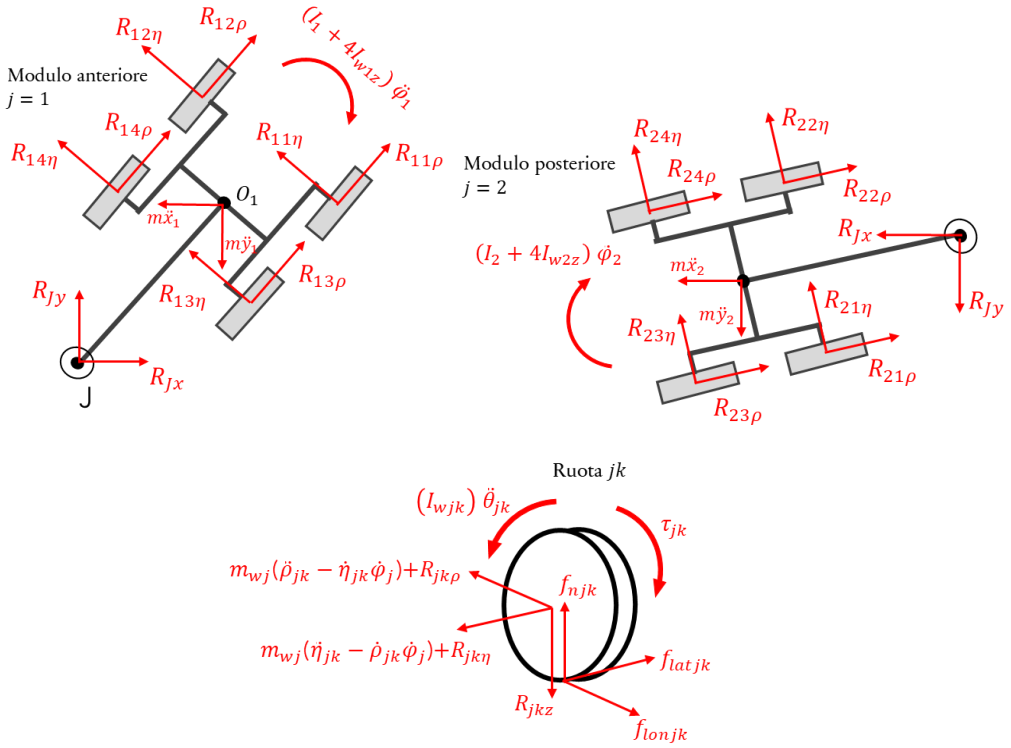

Figura 9.5 – Diagramma di corpo libero dei due moduli e della generica ruota jk

Dai diagrammi di corpo libero in Figura 9.5 è possibile ricavare le tre equazioni che governano il primo (j=1) ed il secondo modulo (j=2):

$$m_j\ddot{x}_j = \cos\varphi_j \sum_{k=1}^{4} R_{jk\rho} - \sin\varphi_j \sum_{k=1}^{4} R_{jk\eta} - (-1)^j R_{Jx} \tag{9.1}$$

$$m_j\ddot{x}_j = \sin\varphi_j \sum_{k=1}^{4} R_{jk\rho} + \cos\varphi_j \sum_{k=1}^{4} R_{jk\eta} - (-1)^j R_{Jy} \tag{9.2}$$

$$(I_j + 4I_{wjz})\ddot{\varphi}_j = b_j \sum_{k=1}^{4} (-(-1)^k R_{jk\rho}) + d_j\left(R_{j1\eta} + R_{j2\eta} - R_{j3\eta} - R_{j4\eta}\right) + L_j(R_{Jx}\sin\varphi_j - R_{Jy}\cos\varphi_j) \tag{9.3}$$

Dove $L_1 = a$ e $L_2 = b$.

Per completare le equazioni dinamiche del sistema, si può definire un insieme di tre equazioni per la generica ruota jk con j=1,2 e k=1,2,3,4. La dinamica verticale viene trascurata

e la forza normale viene utilizzata solo per definire la resistenza al rotolamento. Il valore effettivo della forza normale è lo stesso per tutte le ruote.

$$m_{wj}\ddot{\rho}_{jk} = f_{lonjk} - R_{jk\rho} + m_{wj}\dot{\eta}_{jk}\dot{\varphi}_j \tag{9.4}$$

$$m_{wj}\ddot{\eta}_{jk} = f_{latjk} - R_{jk\eta} - m_{wj}\dot{\rho}_{jk}\dot{\varphi}_j \tag{9.5}$$

$$I_{wj}\ddot{\theta}_{jk} = \tau_{jk} - r_{wj}f_{lonjk} - k_{rollj}f_{njk}\, sign(\dot{\theta}_{jk}) \tag{9.6}$$

Come si può osservare dalle espressioni, la complessità del modello dinamico non risiede nelle equazioni dinamiche o nel loro numero, bensì, come descritto nella sezione successiva, nella modellazione delle forze di contatto, aspetto che non può essere trascurato in quanto la configurazione delle ruote di questi robot fa sì che queste siano caratterizzate da slittamenti laterali importanti.

9.1.1 Modello del contatto ruota terreno

Durante il moto, tutte le ruote della stessa unità di locomozione sono costrette a girare alla stessa velocità a causa della natura del sistema di ingranaggi. Questo porta inevitabilmente ad un fenomeno molto evidente di grande slittamento laterale delle ruote durante le traiettorie curve, anche a bassa velocità.

Nell'introdurre il modello cinematico era stato ipotizzato che ogni ruota rotolasse senza slittare né longitudinalmente né trasversalmente. Tuttavia, a causa dell'elevato numero di punti di contatto, e dei non trascurabili slittamenti a cui sono soggette le ruote, un approccio puramente cinematico difficilmente si adatta al comportamento effettivo di questi robot.

Per modellare correttamente la dinamica di un sistema articolato, le forze di contatto generate dall'interazione ruota-suolo risultano cruciali. La maggior parte degli studi che considerano lo slittamento delle ruote utilizzano la formula magica di Pacejka per modellare le forze di contatto. Purtroppo, il numero di parametri necessari per modellare il contatto è elevato e solitamente non è facile individuarli se le ruote non sono pneumatici commerciali. Per questo motivo viene ripreso il lavoro descritto in [22] in cui viene definito e utilizzato un modello più semplice basato sulla rigidità normale, longitudinale e laterale della ruota.

La forza di contatto normale f_{njk} è definita come una funzione lineare dello spostamento normale della ruota Δz_{jk}:

$$f_{njk} = k_{nj} \cdot \Delta z_{jk} \tag{9.7}$$

In questo modello la dinamica verticale del robot è trascurata; quindi, vale l'ipotesi che la forza normale sia sempre costante, uguale per tutte le ruote e uguale al valore statico. Pertanto, vengono trascurati anche tutti i trasferimenti di carico inerziale.

La forza di contatto longitudinale f_{lonjk} è definita come una funzione lineare della velocità relativa del piano di contatto della ruota, mentre la forza di contatto laterale f_{latjk} è una funzione lineare della velocità laterale della ruota:

$$se \sqrt{{f_{lonjk}}^2 + {f_{latjk}}^2} \leq \mu_s f_{njk} \quad \Rightarrow \quad \begin{cases} f_{lonjk} = k_{lonj}(r_{wj}\dot{\theta}_{jk} - \dot{\rho}_{jk}) \\ f_{latjk} = -k_{latj}\dot{\eta}_{jk} \end{cases} \tag{9.8}$$

$$se \sqrt{{f_{lonjk}}^2 + {f_{latjk}}^2} > \mu_s f_{njk} \quad \Rightarrow \quad \begin{cases} f_{lonjk} = \mu_d f_{njk} \dfrac{k_{lonj}(r_{wj}\dot{\theta}_{jk} - \dot{\rho}_{jk})}{\sqrt{{f_{lonjk}}^2 + {f_{latjk}}^2}} \\ f_{latjk} = -\mu_d f_{njk} \dfrac{k_{latj}\dot{\eta}_{jk}}{\sqrt{{f_{lonjk}}^2 + {f_{latjk}}^2}} \end{cases} \tag{9.9}$$

Alcuni parametri devono essere identificati per validare correttamente il modello proposto. La rigidezza laterale e longitudinale della ruota sono i parametri principali che devono essere identificati mediante l'adattamento dei dati sperimentali. Il processo di stima viene eseguito risolvendo un problema di minimi quadrati non lineare per minimizzare la differenza tra le quantità misurate e gli stati simulati.

Di seguito vengono descritti i test, eseguiti in [22] e [35], effettuati in modo da poter isolare gli effetti dei singoli parametri al fine di semplificare la stima degli stessi. La prima serie di prove consiste nel far muovere il robot in linea retta a velocità diverse per stimare i parametri che influenzano la dinamica longitudinale senza l'influenza di alcuna dinamica laterale. Nella seconda serie di prove, invece, il robot segue una traiettoria circolare con diverse velocità angolari a destra e a sinistra. In questi casi sono presenti dinamiche sia laterali che longitudinali, ma individuando opportunamente quelle longitudinali è possibile focalizzarsi sulla stima di quelle laterali.

La prima serie di test consiste in diverse prove in cui il modulo anteriore del robot viene comandato in modo da raggiungere e mantenere diverse velocità angolari per i due motori che azionano le ruote anteriori producendo traiettorie rettilinee. Da queste prove è possibile identificare i principali parametri che regolano la dinamica longitudinale del sistema: μ_s e μ_d, i coefficienti di attrito statico e dinamico di contatto, k_{lonj}, la rigidezza longitudinale delle ruote, e k_{rollj}, il parametro di attrito volvente. Facendo poi un confronto tra i risultati sperimentali e quelli simulati, è possibile affermare che i risultati sono una buona rappresentazione dei test.

Dati i parametri longitudinali individuati (Tabella 9.2), diventa più facile ottenere una stima delle rigidezze laterali delle ruote k_{latj}, i principali parametri che regolano la dinamica laterale del robot articolato. Al fine di ottenere una buona identificazione, è stato ripetuto più volte un semplice test. In questo test, il motore sul lato destro e quello sul lato sinistro sono stati impostati per mantenere velocità angolari distinte in modo tale da far seguire al robot una traiettoria circolare, con raggio definito. Facendo il confronto tra risultati sperimentali e simulati si ottiene che il raggio medio della traiettoria simulata non si discosta molto dal valore effettivo; questo vuol dire che le grandezze laterali individuate possono essere considerate una buona approssimazione dei risultati sperimentali.

Tabella 9.2 - Parametri stimati

Parametro	*Valore stimato*	*Parametro*	*Valore stimato*
μ_s	0.63	k_{rollj}	$8.32 \times 10^{-3}\ m$
μ_d	0.30	k_{latj}	$2939.81\ Ns/m$
k_{lonj}	$2039.05\ Ns/m$	k_{nj}	$15 \times 10^{3}\ N/m$

9.2 Modello della trasmissione anteriore

Nel modello dinamico descritto nella Sezione 9.1 si ipotizza che le forze normali di contatto siano uguali e costanti.

Passando invece alla descrizione del modello Adams, viene rappresentata la dinamica del robot nella sua completezza: i carichi si distribuiscono opportunamente e viene posta

maggiore attenzione sugli effetti inerziali, svincolandosi così dall'ipotesi di un carico verticale costante ed uguale su ciascuna ruota.

Per mantenere la coerenza con la nomenclatura utilizzata nel modello Adams vengono attuate le seguenti modifiche:

- Il modulo frontale non viene più indicato col pedice 1, ma col pedice F.

$$(x_1, y_1, \varphi_1) \quad \rightarrow \quad (x_\mathrm{F}, y_\mathrm{F}, \varphi_\mathrm{F})$$

- Il modulo posteriore non viene più indicato col pedice 2, ma col pedice R.

$$(x_2, y_2, \varphi_2) \quad \rightarrow \quad (x_\mathrm{R}, y_\mathrm{R}, \varphi_\mathrm{R})$$

La trasmissione viene modellata ripristinando separatamente congruenza ed equilibrio[9]. Per la congruenza, si impone:

$$\omega_{M,F\sim} = \frac{\omega_{R,F\sim F}}{\tau_F} = \frac{\omega_{R,F\sim B}}{\tau_F} \tag{9.10}$$

Tabella 9.3 – Descrizione delle velocità angolari e del rapporto di trasmissione

Simbolo	Descrizione
$\omega_{M,F\sim}$	velocità angolare del motore (M) sinistro (L) o destro (R) del modulo frontale (F)
$\omega_{R,F\sim F}$	velocità angolare della ruota (R) del modulo frontale (F), posta sull'unità motrice di sinistra (L) o destra (R), anteriormente (F)
$\omega_{R,1\sim B}$	velocità angolare della ruota (R) del modulo frontale (F), posta sull'unità motrice di sinistra (L) o destra (R), posteriormente (B)
τ_F	rapporto di trasmissione motore-ruota del modulo frontale

Per l'equilibrio, si applicano ai due lati della trasmissione la coppia trasmessa secondo lo schema sottostante.

[9] I legami funzionali che esistono tra le grandezze cinematiche esterne ed interne vengono chiamati equazioni di compatibilità cinematica o equazioni di congruenza, mentre le relazioni intercorrenti tra le grandezze meccaniche interne ed esterne vengono denotate come equazioni di equilibrio.

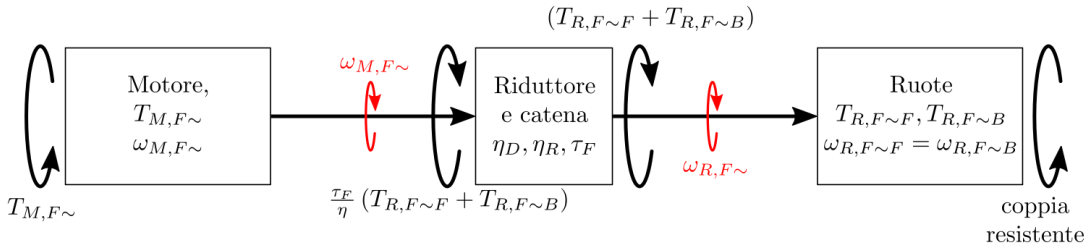

Figura 9.6 – Schema della trasmissione del modulo anteriore

Il rendimento η è funzione del verso del flusso di potenza:

$$T_{M,F\sim}\omega_{M,F\sim} \geq 0 \quad \Rightarrow \quad moto\ diretto \quad \Rightarrow \quad \eta = \eta_D$$
$$T_{M,F\sim}\omega_{M,F\sim} < 0 \quad \Rightarrow \quad moto\ retrogrado \quad \Rightarrow \quad \eta = \frac{1}{\eta_R}$$

Dove per Agri.q $\eta_D = 0.6$ e $\eta_R = 0.35$.

La coppia resistente alle ruote è fornita dal contatto a terra.

9.2.1 Implementazione

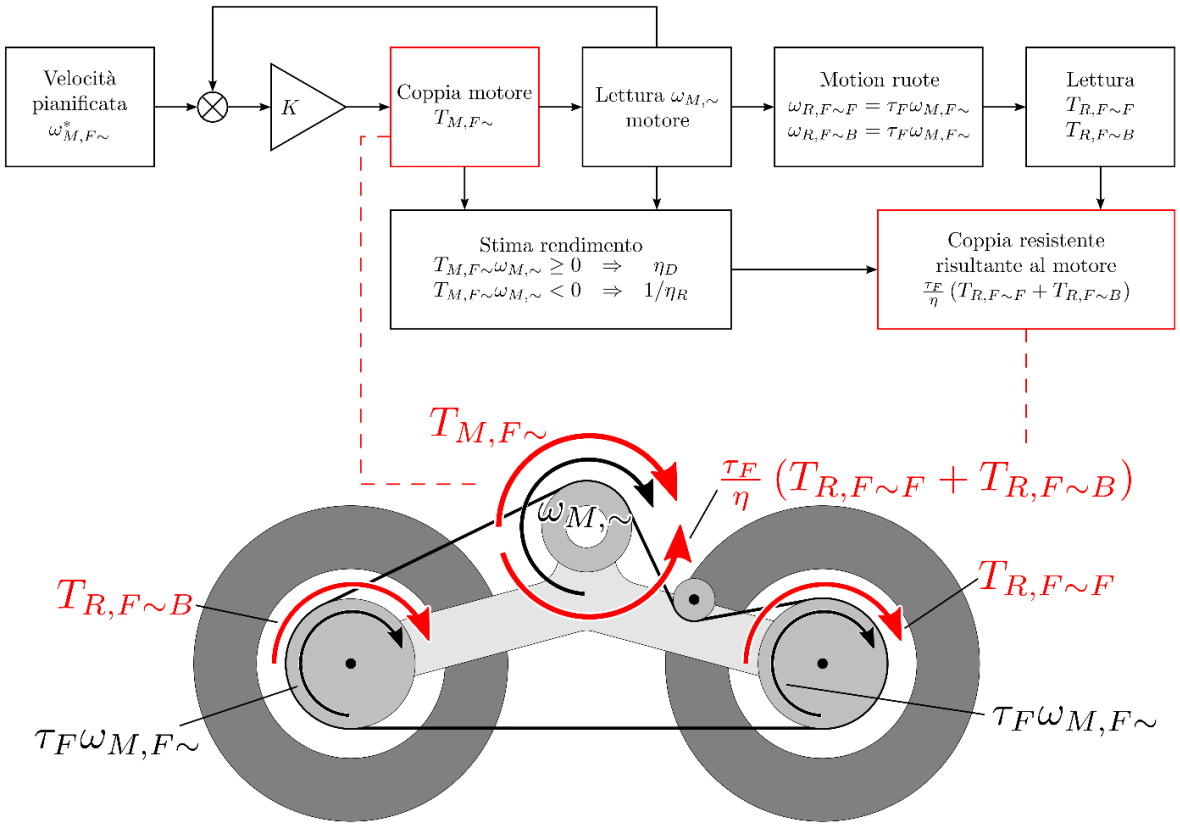

Figura 9.7 – Schema dell'implementazione della trazione anteriore sul modello del robot Agri.q realizzato in Adams

L'implementazione Adams è basata su un driver per ogni ruota che garantisce il rispetto del vincolo $\omega_{M,F\sim} = \tau_F \omega_{R,F\sim F} = \tau_F \omega_{R,F\sim R}$; la coppia generata dal vincolo viene letta ed applicata all'albero motore.

Operativamente (ad esempio per la ruota FRF):

- Si applica un'attuazione al motore FR tramite driver cinematico usando la variabile $\omega_{des\,FR}$ contenente il profilo di velocità desiderato. La velocità angolare del motore FR, $\omega_{des\,FR}$, viene usata come variabile d'ingresso per la co-simulazione Adams/ Matlab.
- Viene generata una misura della velocità angolare del motore a portatreno fermo.
- La misura ottenuta viene imposta tramite driver cinematico alle ruote.
- Si calcola la coppia alla ruota $T_{R,F\sim\sim}$.
- Si stima il verso del flusso dalla potenza[10] istantanea del motore.
- Si applica la reazione risultante da entrambe le ruote all'albero motore.

Nota: la stima del flusso di potenza rende il problema differenziale algebrico. Per evitare il problema, si applica il valore di rendimento stimato agli step di integrazione precedenti introducendo un delay di 1 ms.

9.3 Modello trasmissione posteriore

Nonostante in questa tesi solo le ruote del modulo anteriore siano considerate motrici (non vi è una trazione posteriore), la descrizione della trasmissione del rimorchio non è da trascurare; infatti, il modello di trasmissione del posteriore differisce dall'anteriore in quanto:

- Consiste in un meccanismo a ruota libera il cui funzionamento corrisponde a quello delle bici. Tale peculiarità migliora notevolmente le prestazioni di durata della batteria dell'intero robot poiché evita che i riduttori di trasmissione delle unità posteriori vengano trascinati dal moto del rover quando non sono attivi, con un conseguente miglioramento dell'efficienza. Il meccanismo a ruota libera fa sì che le ruote posteriori di Agri.q possano esercitare una coppia solo in marcia avanti (mentre

[10] In caso di moto diretto il flusso di potenza va dal motore al carico. In caso di moto retrogrado il flusso di potenza va dal carico al motore

le ruote posteriori Epi.q sono in grado di esercitare una coppia in qualsiasi senso di rotazione).

- Il rapporto di trasmissione è differente.

Per la trasmissione posteriore non si può procedere allo stesso modo proprio a causa della natura unilaterale del vincolo di ruota libera:

$$\omega_{M,B\sim} \leq \tau_R \omega_{R,B\sim F} = \tau_B \omega_{R,B\sim B} \tag{9.11}$$

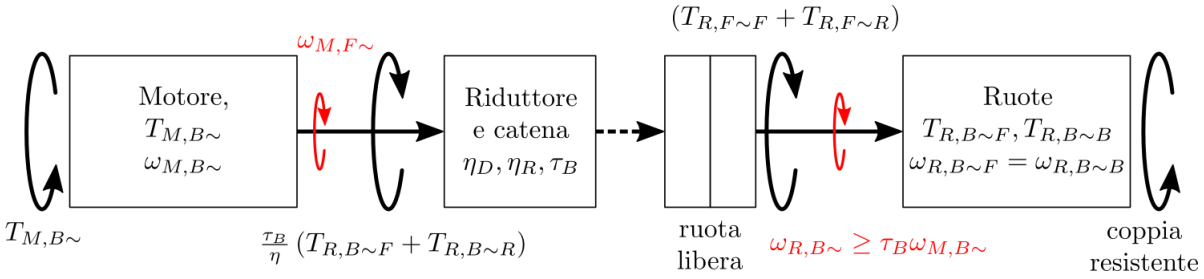

Figura 9.8 – Schema della trasmissione del modulo posteriore

Per applicare la procedura usata per la trasmissione anteriore sarebbe necessario generare un driver cinematico valido solo quando viene applicata una coppia $T_{M,B\sim} > 0$. Il problema si risolve rendendo la coppia trasmessa funzione della velocità angolare, a bilanciere fermo[11], del motore e della ruota. Ovvero:

$$T_{R,B\sim F} = K\left(\omega_{M,B\sim} - \tau_B \omega_{R,B\sim F}\right) \tag{9.12}$$

$$T_{R,B\sim B} = K\left(\omega_{M,B\sim} - \tau_B \omega_{R,B\sim B}\right) \tag{9.13}$$

Con K variabile arbitraria in grado di mantenere il vincolo cinematico entro soglie di errore accettabili.

[11] La trasmissione può essere vista come un meccanismo epicicloidale dove il bilanciere è il portatreno che fa ruotare i satelliti (le ruote). Il pignone è invece la ruota solare. Ipotizzando il bilanciere fermo si semplifica la trattazione. Se invece si considera il bilanciere mobile, la situazione varia, ma solo leggermente, perché l'effetto è presente molto brevemente in fase di accelerazione/decelerazione longitudinale oppure in presenza di ostacoli.

9.3.1 Implementazione

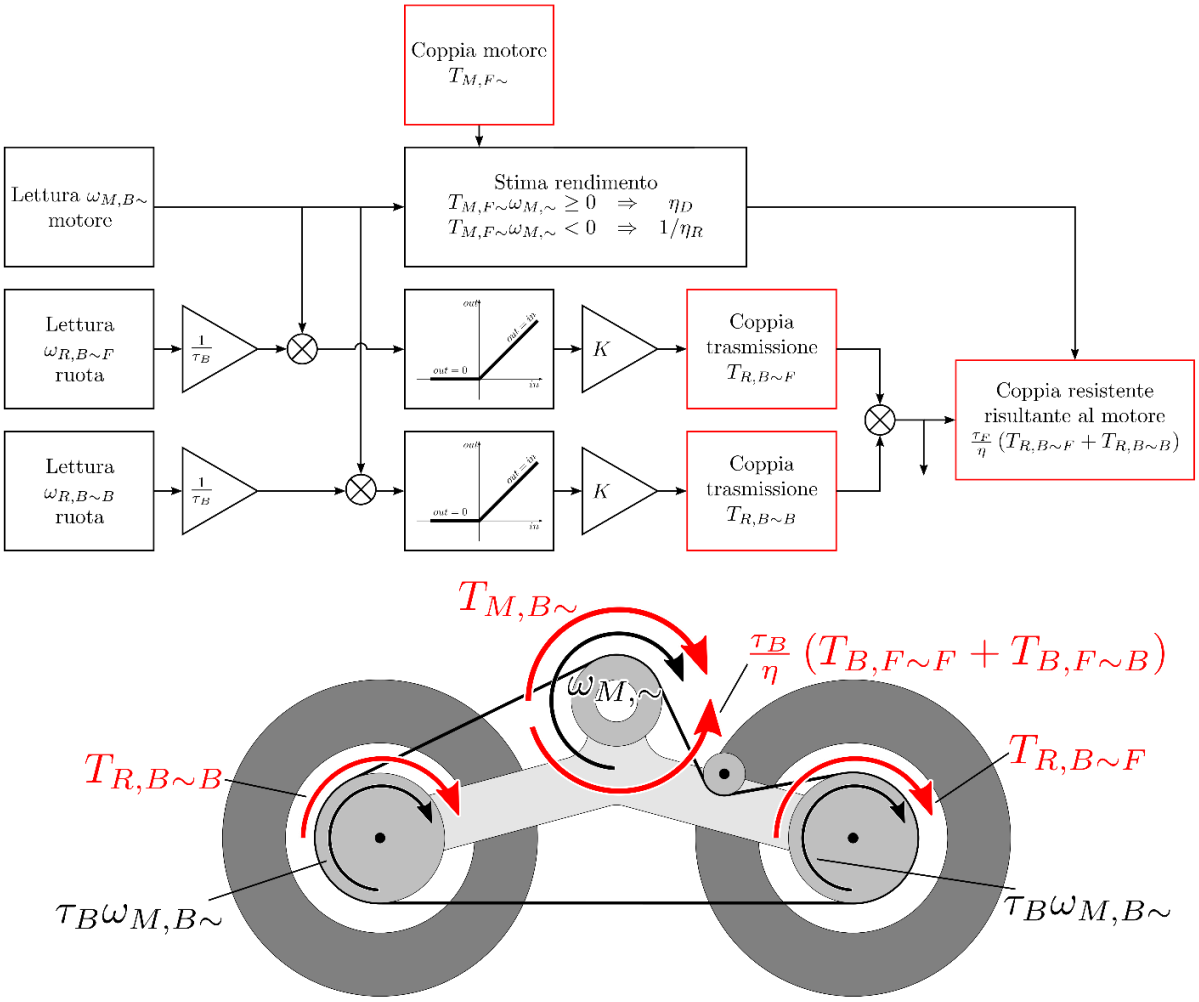

Figura 9.9 – Schema dell'implementazione della trazione posteriore sul modello del robot Agri.q realizzato in Adams

Si procede in maniera analoga alla trasmissione anteriore:

- Viene generata una misura della velocità angolare del motore a portatreno fermo.
- Viene generata una misura della velocità angolare delle ruote a portatreno fermo.
- Si inizializza la coppia motore come variabile di input (da Simulink o come funzione definita dall'utente). Poiché le ruote del modulo posteriore vengono considerate folli si pone questo input, e quindi la coppia motore, uguale a 0.
- Si calcola la differenza di velocità angolare forzandola a valore nullo al di fuori del campo di validità del vincolo di ruota libera.
- Si stima la coppia applicata ad ogni ruota come $T_{R,B\sim F} = K\left(\omega_{M,B\sim} - \tau_B\omega_{R,B\sim F}\right)$ e $T_{R,B\sim B} = K\left(\omega_{M,B\sim} - \tau_B\omega_{R,B\sim B}\right)$.
- Si applicano le coppie $T_{T,B\sim F}$ e $T_{T,B\sim B}$ alle rispettive ruote.

- Si ristabilisce l'equilibrio applicando le stesse coppie all'albero motore, come resistenti alla coppia motrice $T_{M,B\sim}$.

9.4 Saturazione di corrente degli azionamenti

Ciascuno dei 4 azionamenti opera un taglio dell'alimentazione dei motori elettrici per garantire l'erogazione di una corrente superiore a quella nominale per un periodo limitato di tempo. Da manuale, il valore di corrente di picco I_P (cui corrisponde la coppia di picco T_P) può essere mantenuta per $t_P = 2\ s$, trascorsi i quali la corrente assorbita viene riportata a livello nominale I_N (cui corrisponde coppia nominale T_N). In caso di correnti comprese fra quella di picco e quella nominale $I_N < I < I_P$, il tempo di erogazione t può aumentare fino al valore

$$t = \frac{T_P - T_N}{T - T_N} t_P \tag{9.14}$$

Per modellare il fenomeno viene definita la funzione:

$$f(T_{M,\sim}) = \begin{cases} |T_{M,\sim}| \leq T_N & \Rightarrow \quad 0 \\ T_N < |T_{M,\sim}| \leq T_P & \Rightarrow \quad |T_{M,\sim}| - T_N \end{cases}$$

La funzione è nulla quando la coppia erogata è inferiore a quella nominale, mentre è pari all'eccesso di coppia negli altri casi. L'integrale di $f(T_{M,\sim})$ può essere usato per garantire la condizione di saturazione imposta dagli azionamenti, controllando che:

$$\int_0^t f(T_{M,\sim})\, dt \leq (T_P - T_N) t_P$$

In realtà, per essere efficace, l'integrale precedente viene mantenuto sempre maggiore di zero, ovvero si utilizza la definizione di $f(T_{M,\sim})$:

$$f(T_{M,\sim}) = \begin{cases} \left(|T_{M,\sim}| \leq T_N\right) \wedge \left(\int_0^{t-dt} f(T_{M,\sim})\, dt \leq 0\right) & \Rightarrow \quad 0 \\ \left(|T_{M,\sim}| \leq T_N\right) \wedge \left(\int_0^{t-dt} f(T_{M,\sim})\, dt > 0\right) & \Rightarrow \quad |T_{M,\sim}| - T_N \\ \left(T_N < |T_{M,\sim}| \leq T_P\right) & \Rightarrow \quad |T_{M,\sim}| - T_N \end{cases} \tag{9.15}$$

La coppia effettivamente erogata $T^*_{M,\sim}$ viene modificata in funzione del valore dell'integrale:

$$T^*_{M,\sim} = \begin{cases} |T_{M,\sim}| \leq T_N & \Rightarrow \quad T_{M,\sim} \\ T_N < |T_{M,\sim}| \leq T_P \wedge \left(\int_0^{t} f(T_{M,\sim})\, dt < (T_P - T_N)t_P\right) & \Rightarrow \quad T_{M,\sim} \\ T_N < |T_{M,\sim}| \leq T_P \wedge \left(\int_0^{t} f(T_{M,\sim})\, dt = (T_P - T_N)t_P\right) & \Rightarrow \quad T_N \end{cases} \tag{9.16}$$

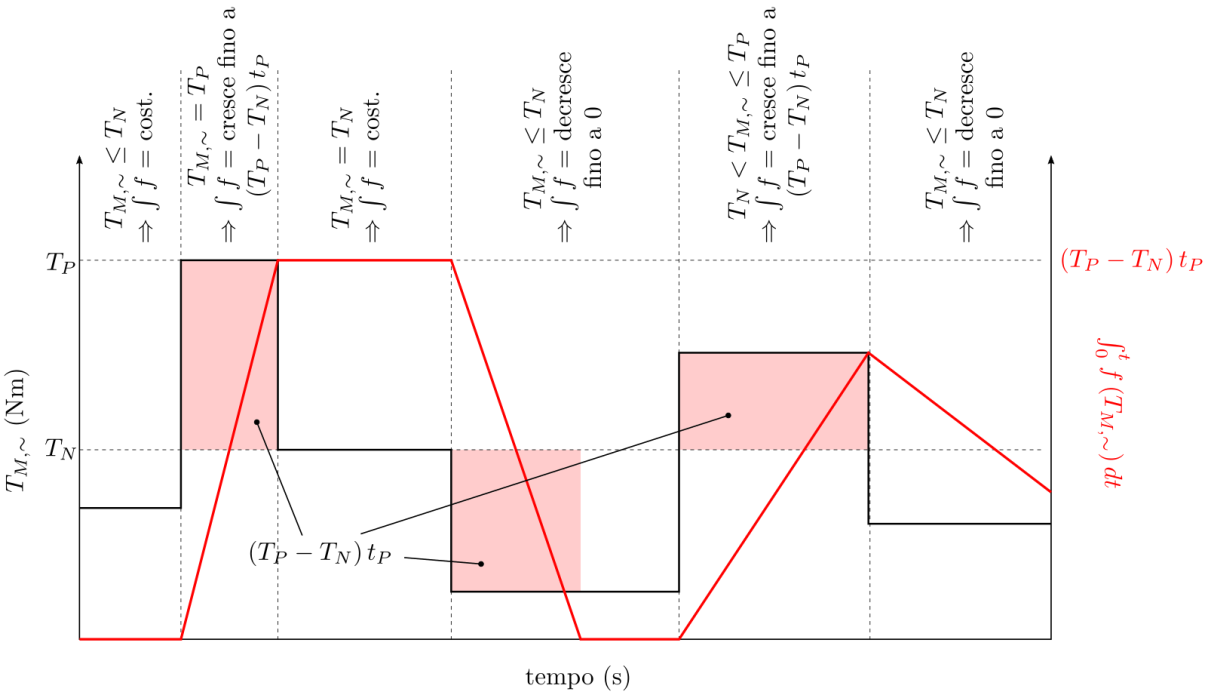

Figura 9.10 – Andamento della coppia erogata $T_{M,\sim}$ (in nero) e dell'integrale della funzione $f(T_{M,\sim})$ (in rosso)

9.4.1 Implementazione

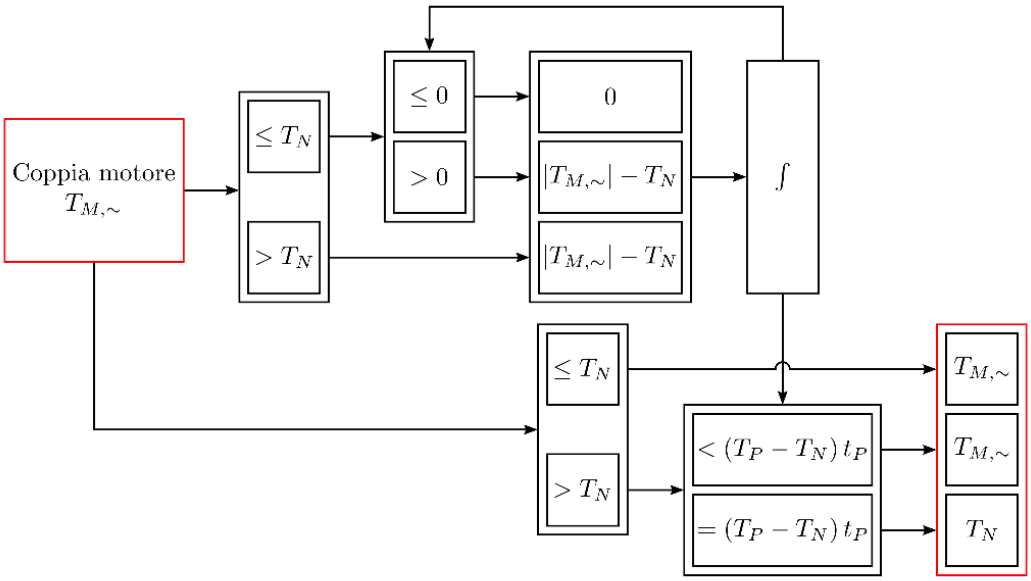

Figura 9.11 – Logica di implementazione con cui viene definita la coppia effettivamente erogata $T^*_{M,\sim}$ dal motore

La coppia effettivamente erogata viene regolata in funzione dell'integrale di $f(T_{M,\sim})$. Il modello di azionamento viene implementato attraverso i seguenti passaggi:

- Per ogni motore, si inizializza la funzione integranda $f(T_{M,\sim})$ e si aggiungono i due IF necessari alla definizione.
- Si introduce una variabile contenente la coppia post-processo di saturazione come da schema precedente.

9.5 Co-simulazione Adams/Matlab

Dopo aver descritto gli elementi più importanti del modello utilizzato si procede con l'attuare la co-simulazione Adams/Matlab in modo da verificare il controllore del moto in retromarcia anche su un modello che non sia unicamente cinematico.

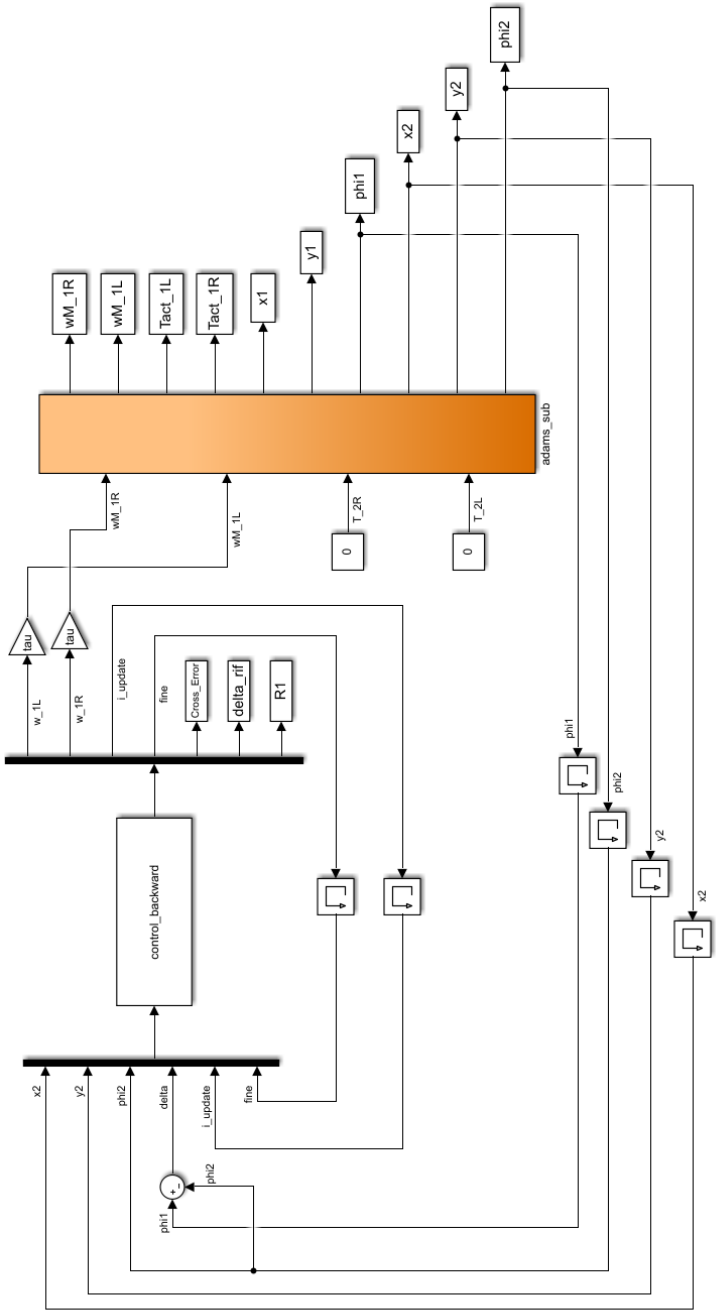

Figura 9.12 – Modello Simulink che racchiude il sistema di controllo (blocco control_backward) ed il modello Adams di Agri.q (blocco adams_sub)

Il controllore della guida autonoma in retromarcia è situato all'interno del blocco 'control_backward'. Questo blocco riceve in ingresso la posa del modulo posteriore (x_2, y_2, φ_2) e l'angolo δ assunti dal robot in ogni istante del moto, più altri due elementi: il primo (i_update) serve all'inseguitore di traiettoria per definire vicino a quale segmento di

percorso si trova il robot, il secondo ($fine$) è di supporto alla definizione degli andamenti di velocità.

In uscita dal blocco di controllo sono presenti:

- Le velocità angolari di riferimento (ω_{1L} , ω_{1R}) da imporre alle ruote sul lato destro e sinistro del modulo frontale. Queste velocità vengono successivamente moltiplicate per il rapporto di trasmissione τ_F in modo da dare in input al modello Adams le velocità angolari del due motori del modulo anteriore ($\omega_{M,FL}$, $\omega_{M,FR}$).
- Il parametro $fine$.
- I parametri ottenuti mediante l'inseguitore di traiettoria, ossia
    - i_update.
    - L'errore laterale ($cross\ error$), cioè la distanza laterale tra il modulo posteriore del robot ed il percorso.
    - Il raggio di curvatura R_1 tale da permettere al modulo anteriore di guidare quello posteriore al fine di raggiungere e mantenere il percorso.

Il modello Adams, oltre a ricevere in input le velocità angolari dei due motori, ha bisogno anche delle coppie relative al modulo posteriore. Tuttavia, poiché si suppone che le ruote del modulo posteriore siano folli, queste coppie vengono poste pari a 0.

Il modello Adams restituisce le seguenti grandezze:

- La velocità angolare a cui effettivamente girano i motori del modulo frontale ($\omega_{M,1L}$, $\omega_{M,1R}$).
- Le coppie erogate dai motori del modulo frontale ($T_{act,1L}$, $T_{act,1R}$).
- La posa del modulo anteriore (x_1, y_1, φ_1).
- La posa del modulo posteriore (x_2, y_2, φ_2).

9.5.1 Risultati ottenuti

Di seguito viene proposto un esempio in cui il sistema di controllo del modo in retromarcia viene prima applicato al modello cinematico del robot Agri.q e successivamente al modello dinamico. In questo modo è possibile vedere come si comporta il controllore nei due casi e fare i dovuti confronti tra gli andamenti ottenuti.

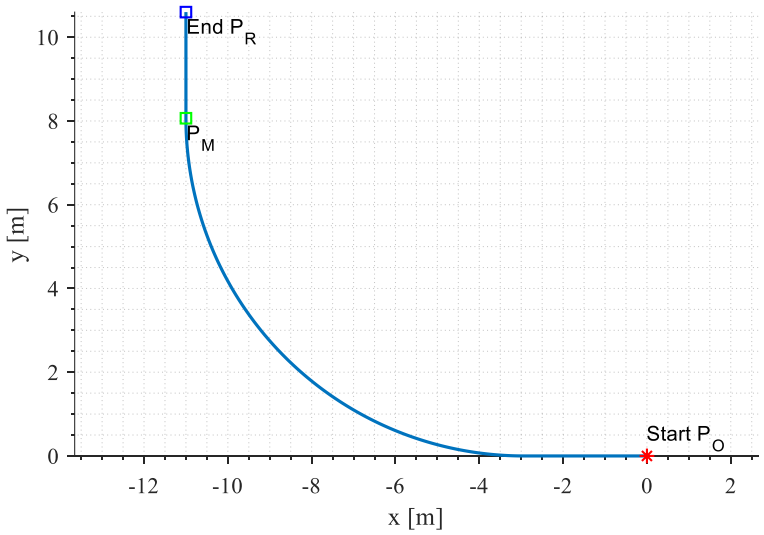

Figura 9.13 – Percorso di riferimento da far percorrere al robot Agri.q in retromarcia

Tabella 9.4 – Grandezze caratteristiche del percorso e della simulazione

	Modello cinematico	**Modello dinamico**	**Unità di misura**
Lunghezza percorso	15.6	15.6	m
Start	[0 0 0 0]	[−1.196 0 0 0]	$m\ m\ deg\ deg$
Goal	[−11 10.6 −90 0]	[−12.196 10.6 −90 0]	$m\ m\ deg\ deg$
Posizione finale modulo posteriore	[−11 10.5 −90.80 −0.56]	[−12.198 10.401 −90.82 −1.24]	$m\ m\ deg\ deg$
Tempo simulazione	0.06	420	s
Accelerazione centripeta massima	0.163	0.116	$\frac{m}{s^2}$
Energia totale richiesta	–	39.61	J

Osservando la Tabella 9.4 si può notare che per il modello cinematico e dinamico è stato utilizzato lo stesso percorso, ma nel caso del modello dinamico questo è stato traslato a sinistra di 1.196 m. Questa traslazione è dovuta al fatto che, nel modello Adams, il punto di riferimento del modulo anteriore (O_1) è sempre posto in corrispondenza dell'origine degli assi del sistema di riferimento globale. In altre parole, nel modello Adams la configurazione iniziale, espressa come spazio delle configurazioni, è sempre $q = [x_1, y_1, \varphi_1, \delta] = [0,0\ ,0,0]$. Riuscire ad avere il modello Adams di Agri.q posizionato o orientato in modo diverso non è semplice e non rappresenta il punto focale di questa tesi. Per questo, al fine di poter paragonare in modo semplice ed immediato il comportamento del robot e la risposta del controllo nel modello cinematico e dinamico, si agisce sul percorso traslandolo.

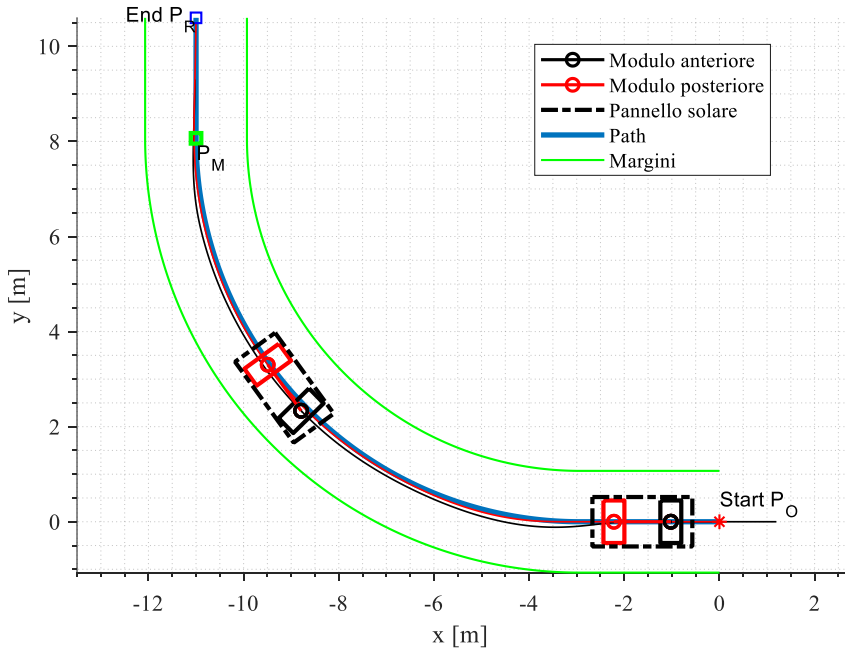

Figura 9.14 – Rappresentazione del moto in retromarcia di Agri.q basato sul modello cinematico del robot

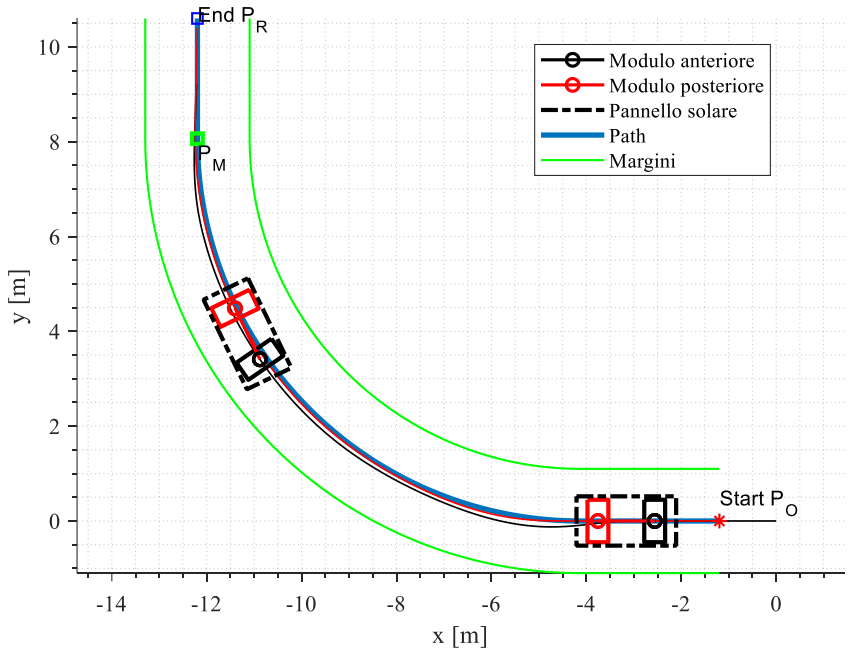

Figura 9.15 – Rappresentazione del moto in retromarcia di Agri.q basato sul modello dinamico del robot

Dall'analisi delle Figure 9.14 e 9.15 non si osserva una variazione apprezzabile del comportamento del robot passando da modello cinematico a dinamico. In realtà si ha un

aumento dell'errore laterale quando si prede in considerazione il modello dinamico del robot, come visibile in Figura 9.16. Questo aumento dell'errore è dovuto al fatto che, poiché Agri.q è un veicolo sottosterzante[12], il robot tende ad allargare maggiormente le curve, discostandosi dal percorso in misura maggiore rispetto a quanto succedeva col solo modello cinematico. Maggiore è la curvatura del percorso più il distacco tra i due modelli diventa evidente, perché per raggi di curvatura più piccoli il fenomeno dello slittamento laterale delle ruote diventa sempre più preponderante.

Anche se nel modello dinamico l'errore laterale aumenta questo non comporta uno scostamento eccessivo del robot dal percorso, infatti l'ingombro del robot (ed in particolare il pannello solare) rimane sempre all'interno dei margini (in verde). Si ricorda che la distanza tra i due margini è pari a 2.1 m.

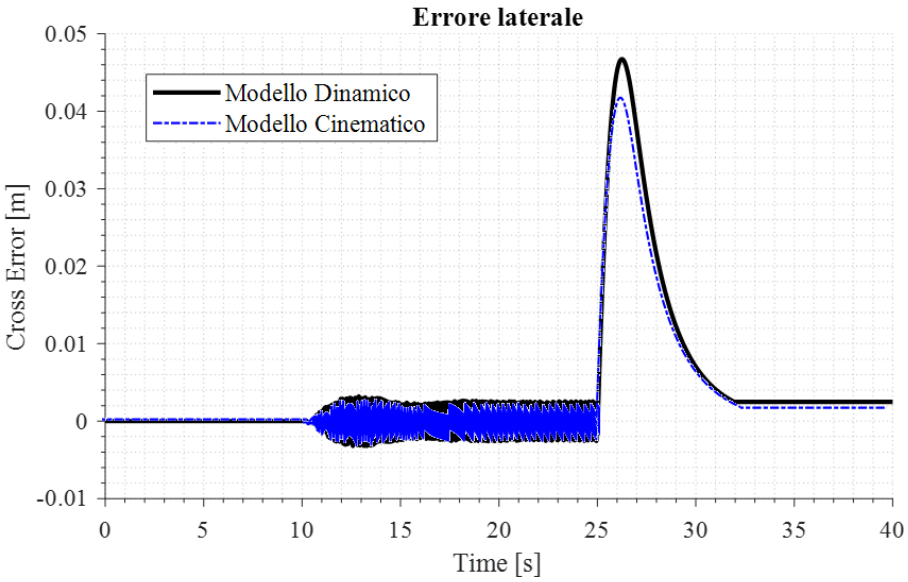

Figura 9.16 – Andamento dell'errore laterale nel tempo, confronto tra risultati ottenuti con modello cinematico e dinamico

[12] Il comportamento sottosterzante non è dovuto solo alla dinamica del veicolo, ma anche alla presenza del controllore del moto in retromarcia. È possibile individuare se il veicolo è sottosterzante o meno solo quando il rimorchio è controllato, cioè quando il veicolo è stabilizzato. Se il veicolo si muove all'indietro e non è presente il controllo, il fenomeno del jackknifing non permette di determinare se il veicolo sia sottosterzante o meno.

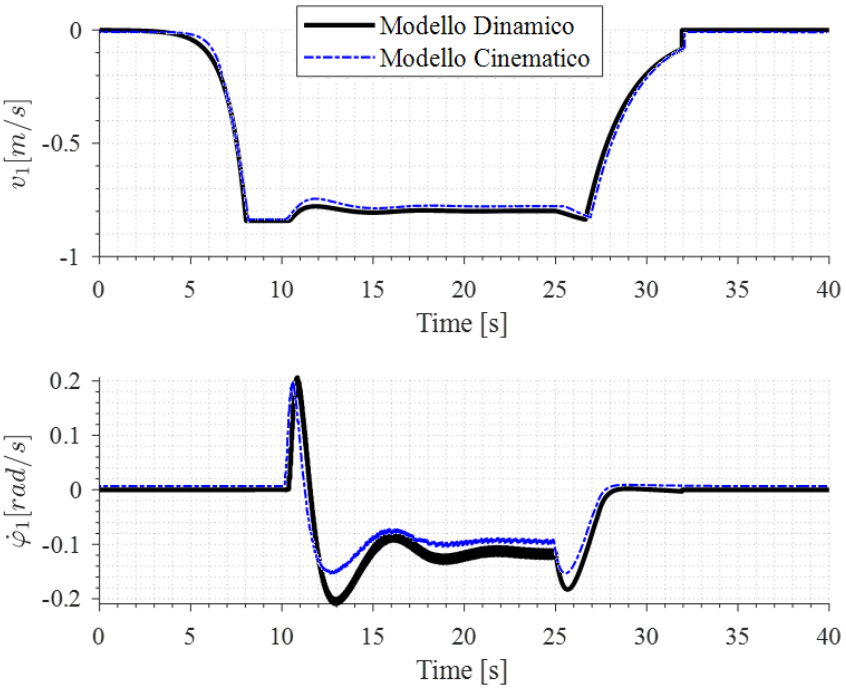

Figura 9.17 – Andamento della velocità longitudinale ed angolare del modulo anteriore nel tempo, confronto tra risultati ottenuti considerando modello cinematico o dinamico

Confrontando gli andamenti tra le velocità in Figura 9.17 si nota un comportamento oscillatorio più marcato della velocità angolare nel caso del modello dinamico. Ciò è dovuto al fatto che gli slittamenti laterali delle ruote portano ad una risposta più aggressiva da parte del sistema di controllo del moto in retromarcia (a parità di guadagni K_P e K_D del controllore e di lunghezza di lookahead L). Un comportamento analogo si può osservare considerando le velocità angolari delle ruote del modulo anteriore (Figura 9.18).

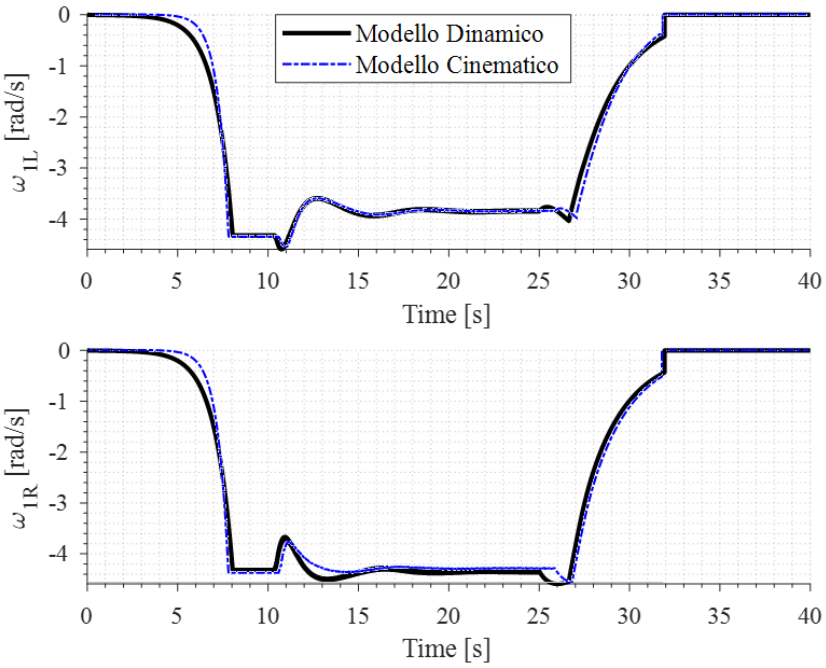

Figura 9.18 – Andamento delle velocità angolari delle ruote del modulo anteriore nel tempo, confronto tra risultati ottenuti considerando modello cinematico o dinamico

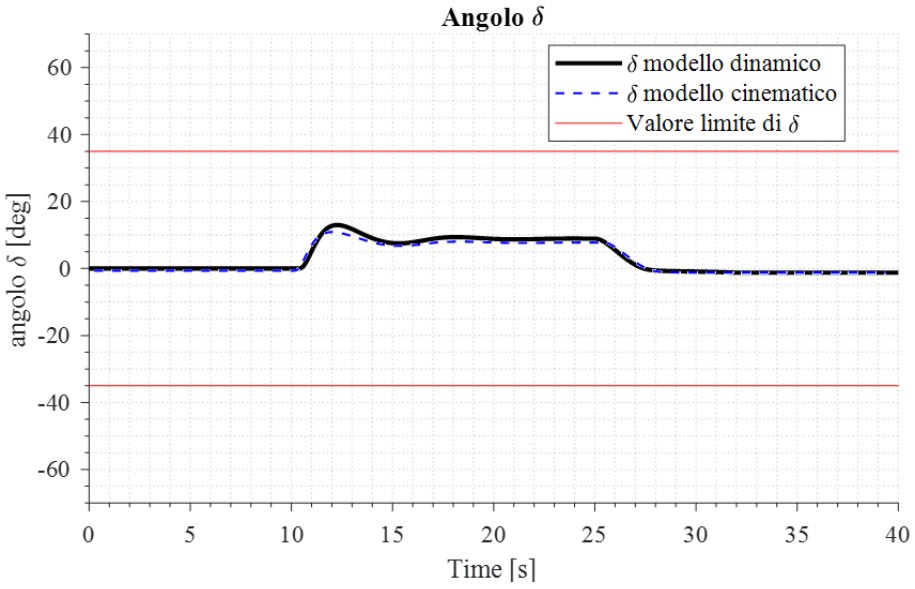

Figura 9.19 – Andamento dell'angolo δ nel tempo, confronto tra risultati ottenuti con modello cinematico e dinamico

Osservando la Figura 9.19 si può vedere come il controllore del modo in retromarcia resti valido passando dal modello cinematico al modello dinamico in quando l'andamento dell'angolo δ è sempre compreso fra $(-\delta_{max}, \delta_{max})$.

Un ulteriore aspetto che si può considerare quando si studia il modello dinamico del robot riguarda la coppia erogata dai motori destro e sinistro del modulo frontale.

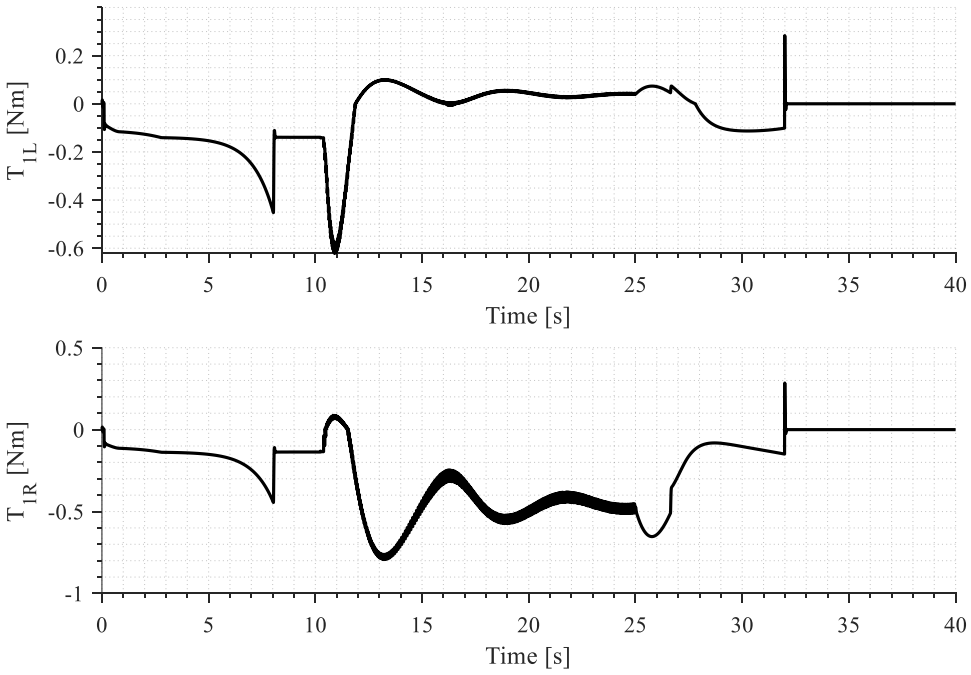

Figura 9.20 – Andamento temporale delle coppie erogate dal motore sinistro e destro del modulo anteriore

Un primo aspetto che si può notare guardando la Figura 9.20 è che, all'oscillazione della velocità angolare $\dot{\varphi}_1$, è associato il comportamento oscillante della coppia erogata dai motori, visibile soprattutto nella coppia T_{1R} del motore destro, indice di una correlazione tra le coppie e la velocità angolare. Infatti, il controllo dell'angolo δ è eseguito agendo sulla velocità angolare $\dot{\varphi}_1$ che a sua volta è generata dalle coppie T_{1R} e T_{1L}. Questa oscillazione non si nota nell'andamento della velocità longitudinale in quanto questa velocità non rientra nel controllo dell'angolo δ.

La banda di oscillazione all'interno della coppia è tuttavia un elemento negativo in quanto implica un maggiore stress del motore. Per quanto si possa cercare di ridurre questo fenomeno, è impossibile annullarlo completamente poiché è dovuto alla natura cinematica del fenomeno del jackknifing che costringe il controllore dell'angolo δ a stabilizzare tale angolo attorno ad una condizione di equilibrio instabile.

La discontinuità finale della coppia è frutto del gradino di velocità finale dovuto al sistema di controllo che arresta il robot una volta che il modulo posteriore si trova

abbastanza vicino al punto finale P_R. Questa discontinuità si potrebbe rimuovere andando a rivedere la gestione dell'avvicinamento del robot all'ultimo waypoint.

Analizzando gli andamenti di coppia si nota anche un comportamento particolare dei motori di Agri.q dovuto alle scelte progettuali: per curve sufficientemente strette, il motore lato interno curva (in questo esempio il motore sinistro) genera una coppia nulla o molto ridotta[13], mentre il motore esterno curva (in questo esempio il destro) si prende carico di tutta la manovra tanto da essere molto vicino al valore limite nominale di 0.5 Nm (vengono raggiunte anche coppie superiori, fino ad 1 Nm, ma per un breve intervallo tempo, poi interviene la protezione, descritta nella Sezione 9.5, che riporta la coppia al valore nominale).

Facendo la media tra le due coppie si può avere una stima di quanta coppia sia necessaria al moto longitudinale (Figura 9.21 in nero). Facendo la differenza tra le coppie si ottiene invece un'indicazione su quanta coppia sia necessaria al moto di imbardata (Figura 9.21 in rosso).

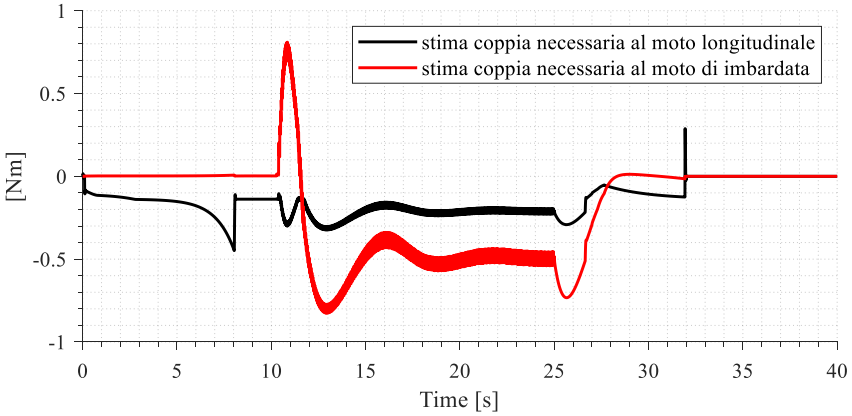

Figura 9.21 – Stima della coppia necessaria al moto longitudinale ed angolare

Quello che si nota, e ci si aspetta, è che, per la maggior parte del moto, il contributo longitudinale è inferiore a quello di imbardata proprio perché l'architettura delle unità di locomozione porta e delle importanti componenti di velocità laterale a terra.

Nella Tabella 9.4 viene anche riportato, nel caso dinamico, il valore dell'energia totale richiesta dai motori, utile per determinare il consumo della carica delle batterie che alimentano i motori e per avere un'indicazione sull'autonomia del robot Agri.q.

[13] Nelle simulazioni a volte il motore lato interno curv a gira al contrario (nel grafico cambia il segno della coppia), nella realtà la componente di segno opposto è data da effetti di contatto e di trasmissione non facilmente modellabili.

Nella Figura 9.22 viene riportato il modo del robot visto in ambente Adams. Confrontando le Figure 9.15 e 9.22 si può vedere come il moto del robot in retromarcia rappresentato mediante Matlab corrisponda a quello rappresentato in Adams. Nelle Figure 9.23 e 9.24 viene raffigurata la configurazione del robot nell'istante di tempo $t = 23.985\ s$ in modo da poter apprezzare la diversa orientazione dei due moduli durante la curva.

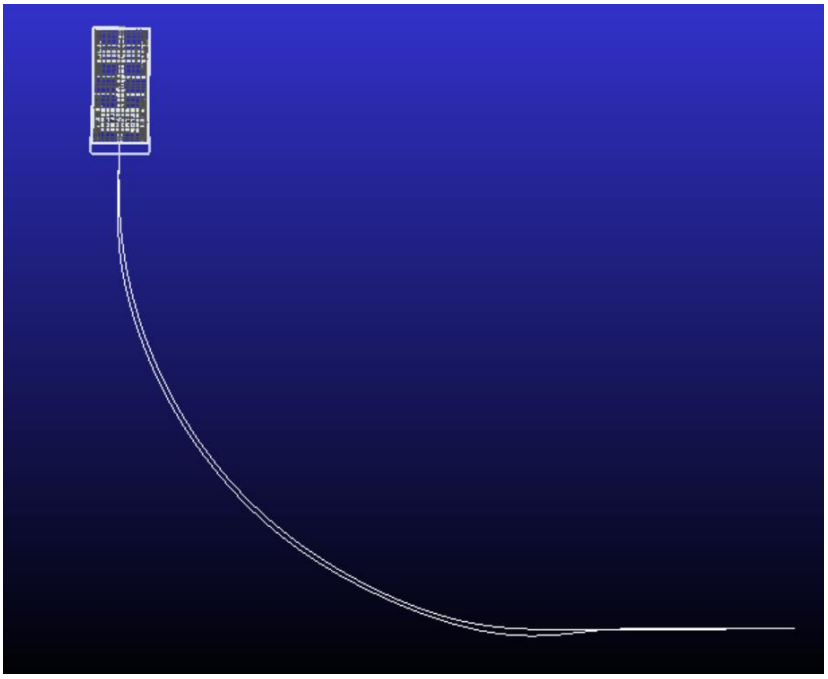

Figura 9.22 – Moto in retromarcia del robot Agri.q visto in Adams

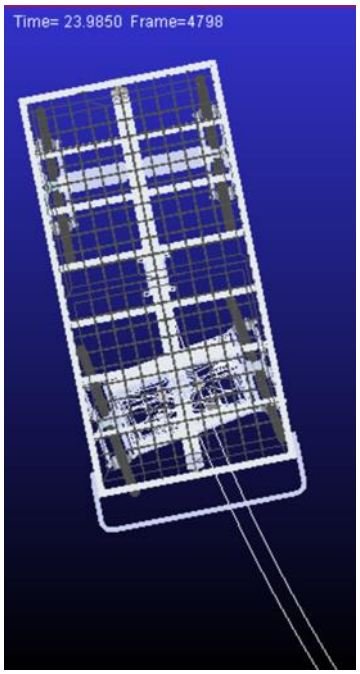

Figura 9.23 – Rappresentazione di Agri.q nell'istante di tempo $t = 23.985\ s$, vista dall'alto

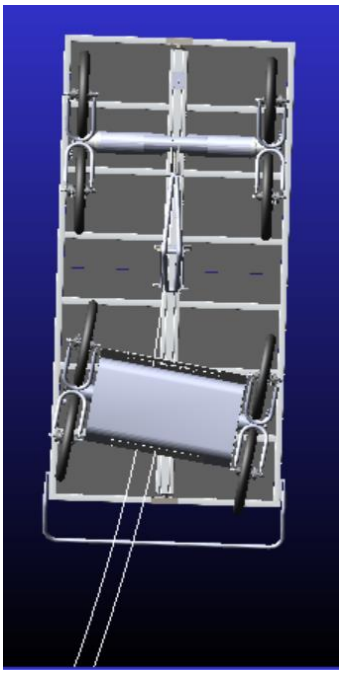

Figura 9.24 – Rappresentazione di Agri.q nell'istante di tempo $t = 23.985\ s$, vista dal basso

10. Capitolo 10: Conclusioni

Con la presente tesi è stato creato un algoritmo per il controllo del moto in retromarcia dei robot mobili articolati Epi.q ed Agri.q, con l'obiettivo specifico di evitare il fenomeno del jackknifing, un fenomeno tipico che riguarda tutti i veicoli articolati e che nelle manovre in retromarcia può verificarsi anche a bassa velocità. Il jackkinfe porta alla divergenza dell'angolo δ (hitch angle) tra motrice e rimorchio rendendo il veicolo incontrollabile.

Il sistema di controllo realizzato si basa sul seguente principio: mentre nel caso di moto in marcia avanti si prende come riferimento il modulo anteriore, in caso di moto in retromarcia, per non perdere il controllo del rimorchio, è necessario avere come riferimenti la posa del modulo posteriore ed il suo raggio di curvatura. Non a caso il controllore viene accompagnato da un algoritmo per la definizione del percorso in marcia indietro riferito al rimorchio. Questo percorso viene elaborato una sola volta all'inizio della missione, a veicolo fermo, con l'obiettivo di portare il modulo posteriore da una configurazione iniziale ad una configurazione finale. La generazione di un percorso da effettuare in retromarcia deve rispettare determinate caratteristiche poiché la presenza del rimorchio, che comporta un vincolo sulla sua rotazione massima rispetto al modulo anteriore, richiede una particolare attenzione per quanto riguarda la scelta delle traiettorie con esso compatibili. A questo scopo sono state create delle funzioni di previsione, in grado di prevedere l'angolo, assunto dal rimorchio, al termine di determinate porzioni di traiettoria, quando il robot si muove all'indietro. Il processo di definizione del percorso ottimale deve essere calibrato in modo da ottenere il giusto compromesso tra lunghezza della traiettoria e rispetto dei limiti del hitch angle tra i moduli.

Dopo aver definito il percorso in uno spazio libero si è passati alla ricerca di un percorso in grado di portare il robot da un punto all'altro in marcia indietro, senza che questo andasse ad urtare un ostacolo. Per ottenere questo risultato è stato utilizzato l'algoritmo Probabilistic RoadMap, un algoritmo per la definizione della successione di punti che permette di passare da un punto ad un altro evitando eventuali ostacoli presenti sulla mappa. Questo algoritmo è stato scelto perché il suo funzionamento si compone di due fasi distinte: nella prima, che è anche la più lunga, si ottiene la rete di nodi che identificano lo spazio di lavoro del robot libero da impedimenti, nella seconda fase, la più breve, nota questa rete vengono selezionati

i nodi in modo da andare dal punto iniziale a quello finale. Questo sistema risulta particolarmente adatto per il robot Agri.q: prendendo come esempio l'applicazione nelle vigne si può pensare di calcolare la prima fase una sola volta, ottenendo una fitta rete di punti, e poi, a seconda della tabella di marcia del robot, andare a ripetere la seconda fase più volte senza che vi sia un elevato tempo di calcolo, in quanto la prima fase è già stata calcolata. Considerazioni analoghe possono essere fatte per Epi.q.

Il percorso così ottenuto deve essere mantenuto come un riferimento da seguire per tutto il tempo, fino alla fine della corsa. La capacità del robot di raggiungere e mantenere la traiettoria prefissata è possibile grazie ad uno specifico algoritmo di inseguimento del percorso, noto come 'Inseguimento Puro'. Mediante questo algoritmo è possibile stabilire il raggio di curvatura che il modulo posteriore deve seguire per poter mantenere il percorso. Noto il raggio di curvatura richiesto al modulo posteriore si può ricavare, sfruttando delle relazioni geometriche, il corrispondente raggio di curvatura del modulo anteriore del robot articolato. Questo passaggio è fondamentale poiché è stato ipotizzato che solo le ruote del modulo anteriore siano motrici; quindi, è necessario conoscere il raggio di curvatura del modulo anteriore per poter definire i riferimenti di velocità da dare al modello cinematico, comune ai due robot, al fine di manovrare opportunamente il modulo posteriore. I riferimenti di velocità sono rappresentati dalle velocità angolari che si vogliono applicare alle ruote del lato destro e sinistro del modulo frontale.

Benché questo ragionamento valga per entrambi i robot, la particolare configurazione del robot Agri.q porta alla necessità di aggiungere un ulteriore sistema di controllo. È stato pertanto sviluppato un controllore che sia in grado di stabilizzare l'angolo δ attorno ad un valore di riferimento. Tale valore di riferimento corrisponde all'angolo di equilibrio che è tipico della tipologia di traiettoria (rettilineo o curvilinea) percorsa in quell'istante. L'instabilità di questo angolo nel moto in marcia indietro rappresenta la spiegazione analitica del fenomeno del jackknifing. Il supporto di questo ulteriore controllo permette al robot Agri.q di seguire il percorso in retromarcia senza che il rimorchio si richiuda sulla motrice.

Una volta dimostrata la validità del controllore del moto in retromarcia sul modello cinematico dei robot, si è passati alla sua verifica nel caso del modello dinamico; nello specifico il controllore è stato verificato su un modello Adams del rover Agri.q. Tale analisi

risulta indispensabile in quanto il modello cinematico si basa unicamente su una visione 2D del sistema e non considera alcune caratteristiche che hanno un peso importante nel comportamento del robot. Tra queste si ricordano: il fatto che vi sono 8 ruote a contatto col terreno, gli slittamenti laterali a cui sono soggette le ruote e la saturazione di corrente degli attuatori.

Analizzando i risultati ottenuti dal controllo sul modello dinamico è stato possibile concludere che, nel passaggio tra il modello cinematico e quello dinamico, si assiste ad un leggero aumento dell'errore laterale tra la posizione del robot e quella del percorso. Nonostante questo aumento dell'errore laterale è stato possibile affermare che il controllo proposto risulta valido anche nel caso del modello dinamico.

È importante evidenziare che il sistema di controllo, benché sia stato pensato per due robot specifici, rappresenta uno strumento di ben più ampia portata: avendo basato il controllo sul modello cinematico dei robot, esso mantiene la sua validità per tutti i veicoli che presentano uno schema cinematico simile ad uno dei due rover. Il sistema sviluppato potrebbe essere esteso ad esempio ad alcune macchine per la movimentazione della terra.

Una volta definita la migliore soluzione per un veicolo articolato in cui solo le ruote del modulo frontale sono motrici, mentre il modulo posteriore viene semplicemente trascinato (o per meglio dire spinto se si considera il moto all'indietro), le fasi future della ricerca potrebbero indagare sull'efficacia di layout alternativi di robot mobili, ad esempio considerando quattro unità di locomozione azionabili (invece di due attive e due inattive). In questo modo si potrebbe avere un controllo diretto sia direzionale che longitudinale del posteriore del robot Epi.q perché, potendo controllare sia i motori dell'avantreno che del retrotreno, si avrebbe un controllo diretto sia sull'angolo δ tra i moduli sia sulla corrispondente velocità $\dot{\delta}$. Una possibile strategia potrebbe essere quella di far gestire al modulo anteriore la traiettoria, sia in avanti che indietro, mentre il posteriore, avendo la possibilità di sterzare da solo, avrebbe il compito di mantiene la stabilità. Inoltre, considerando l'azione del controllo sul modello dinamico di Epi.q, si potrebbe controllare se, applicando opportunamente la coppia al posteriore, è possibile aumentare la stabilità del robot. Con Agri.q questo approccio non è contemplato poiché, anche introducendo la trazione posteriore, non si potrebbe avere il controllo diretto del rimorchio, ma soltanto indiretto; quindi, non si avrebbe un vantaggio rilevante in termini di controllore. Il controllo

diretto del modulo posteriore di Agri.q non è possibile in quanto le ruote posteriori non possono trasmettere coppia all'indietro essendo applicato un meccanismo a ruota libera come quello delle biciclette.

Un altro possibile campo di sviluppo potrebbe riguardare il calcolo del percorso. In questa tesi il percorso di riferimento utilizzato è stato il percorso di Dubins, il quale ha come limite il fatto di poter raggiungere il punto finale potendo procedere in una sola direzione, in questo caso solo all'indietro. Il passo successivo dovrebbe essere quello di considerare la capacità del robot di poter andare in entrambe le direzioni e quindi di poter eseguire manovre; sarebbe opportuno considerare come si comporta il controllore in questo caso.

In conclusione, è possibile affermare che il sistema di controllo del moto in retromarcia sviluppato in questa tesi per i robot Epi.q ed Agri.q rappresenta un'ottima base su cui poter sviluppare controllori più complessi.

Bibliografia

[1] Murugendran, B., Transeth, A. A., and Fjerdingen, S. A. (2009). Modeling and path-following for a snake robot with active wheels. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, IEEE (pp. 3643–3650).

[2] Tanaka, M., Tanaka, K., and Matsuno, F. (2014). Approximate path-tracking control of snake robot joints with switching constraints. In *IEEE/ASME transactions on mechatronics*, 20(4), (pp. 1633–1641).

[3] Tanaka, M., and Tanaka, K. (2016). Singularity analysis of a snake robot and an articulated mobile robot with unconstrained links. In *IEEE Transactions on Control Systems Technology*, 24(6), (pp. 2070–2081).

[4] Allen, T. J., Quinn, R. D., Bachmann, R. J., & Ritzman, R. E. (2003, October). Abstracted biological principles applied with reduced actuation improve mobility of legged vehicles. In *Proceedings of the 2003 IEEE/RSJ - Int. Conference on Intelligent Robots and Systems*, Las Vegas, Nevada.

[5] Schroer, R. T., Boggess, M. J., Bachmann, R. J., Quinn, R. D., & Ritzmann, R. E (2004, May). Comparing cockroach and whegs robot body motions. In *Proceedings of ICRA 2004 - International Conference on Robotics and Automation*, Las Vegas, Nevada.

[6] Quaglia, G., Bruzzone, L., Oderio, R., & Razzoli, R. P. (2011, January). Epi. Q Mobile robots family. In *ASME International Mechanical Engineering Congress and Exposition* (Vol. 54938, pp. 1165-1172).

[7] Dhaouadi, R., Hatab, A.A. (2013). Dynamic modelling of differential-drive mobile robots using lagrange and newton-euler methodologies: A unified framework. In *Advances in Robotics & Automation* (Vol 2, pp. 1–7).

[8] Laumond, J-P., Jacobs, P.E., Taix, M., Murray, R.M. (1994). A motion planner for non-holonomic mobile robots. In *IEEE Transactions on Robotics and Automation* (Vol.10, pp. 577–593).

[9] Sampei, M., Tamura, T., Kobayashi, T., & Shibui, N. (1995). Arbitrary path tracking control of articulated vehicles using nonlinear control theory. In *IEEE Trans. on Control System Technology* (Vol. 3, no. 1, pp. 125–131).

[10] DeSantis, R.M. (1998). Path-tracking for articulated vehicles via exact and jacobian linearization. In *IFAC Intelligent Autonomous Vehicles* (pp. 159–164).

[11] Bolzern, P., DeSantis, R. M., Locatelli, A., & Masciocchi, D. (1998). Pathtracking for articulated vehicles with off-axle hitching. In *IEEE Trans. on Control System Technology* (Vol. 6, no. 4, pp. 515–523).

[12] Gonz´alez-Cantos, A., Ollero, A. (2009). Backing-up maneuvers of autonomous tractor-trailer vehicles using the qualitative theory of nonlinear dynamical systems. In *The Int. J. of Robotics Research* (Vol. 28, no. 1, pp. 49–65).

[13] Beglini, M., Lanari, L., Oriolo, G. (2020). Anti-Jackknifing Control of Tractor-Trailer Vehicles via Intrinsically Stable MPC. In *IEEE International Conference on Robotics and Automation (ICRA)*, Paris, France.

[14] Petrov, P. (2010). Nonlinear backward tracking control of an articulated mobile robot with off-axle hitching. In *Proc. WSEAS Int. Conf. ISPRA* (pp. 269–273).

[15] He, Y., Ren, J. (2013). A Comparative Study of Car-Trailer Dynamics Models. In *SAE International Journal of Passenger Cars-Mechanical Systems* (pp. 177–186).

[16] Scheding, S., Dissanayake, G., Nebot, E.M., Durrant-Whyte, H. (1999). An experiment in autonomous navigation of an underground mining vehicle. In *IEEE Transactions on Robotics and Automation* (pp. 85–95).

[17] Pradalier, C., Usher, K. (2007). A simple and efficient control scheme to reverse a tractor-trailer system on a trajectory". In *Proceedings 2007 IEEE International Conference on Robotics and Automation* (pp 2208–2214), CSIRO ICT Centre, Autonomous Systems Laboratory, Brisbane, Australia, DOI: 10.1109/ROBOT.2007.363648.

[18] Lamiraux, F., Laumond, J. P. (1998). A practical approach to feedback control for a mobile robot with trailer. In *IEEE Int. Conf. On Robotics and Automation* (pp. 3291–3295).

[19] Gherlone, G. (2002, May). Algoritmo generatore di traiettorie per un veicolo autonomo con rimorchio. Master's thesis, mechanical engineering, Politecnico di Torino, Italy.

[20] Bruzzone, L., Quaglia, G. (2012). Locomotion systems for ground mobile robots in unstructured environments. *Mechanical sciences* (Vol. 3, pp. 49-62).

[21] Bozzini, G., Bruzzone, L., Oderio, R., Quaglia, G., & Razolli, R. (2009). Design of the small mobile robot Epi. q-2. *Proceedings of AIMETA*.

[22] Botta, A., Cavallone, P., Carbonari, L., Tagliavini, L., & Quaglia, G. (2021). Modelling and Experimental Validation of Articulated Mobile Robots with Hybrid Locomotion System. In *Advances in Italian Mechanism Science* (pp. 758–767). Cham, DOI: 10.1007/978-3-030-55807-9_84.

[23] Quaglia, G., Oderio, R., Bruzzone, L., & Razzoli, R. (2013). A modular approach for a family of ground mobile robots. *International Journal of Advanced Robotic Systems*, *10*(7), 296.

[24] Bruzzone, L., Oderio, R., Quaglia, G., & Razzoli, R. (2010). Experimental assessment and evolution perspectives of the Epi.q mobile robot architecture. *International Journal Of Robotics Research* (pp. 81-91).

[25] Vecchio, Y., De Rosa, M., Adinolfi, F., Bartoli, L., & Masi, M. (2020). Adoption of precision farming tools: a context-related analysis. Land use policy. 94, 104481.

[26] Balafoutis, A., Beck, B., Fountas, S., Vangeyte, J., Van Der Wal, T., Soto, I., Gómez-Bar-bero, M., Barnes, A., & Eory, V. (2017). Precision agriculture technologies positively contrib-uting to GHG emissions mitigation, farm productivity and economics. *Sustainability* (Vol.9, pp. 1339).

[27] Chatzimichali, A. P., Georgilas, I. P., Tourassis, V.D. (2009). Design of an advanced prototype robot for white asparagus harvesting. In *International Conference on Advanced Intelligent Mechatronics* (pp. 887-892). Singapore.

[28] Aljanobi, A. A., Al-Hamed, S. A., & Al-Sushaibani, S. A. (2010). A setup of mobile robotic unit for fruit harvesting. In *19th International Workshop on Robotics in Alpe-Adria-Danube Region RAAD*, Budapest, Hungary.

[29] Tokekar, P., Hook, J. V., Mulla, D., & Isler, V. (2016). Sensor planning for a symbiotic UAV and UGV system for precision agriculture. In *IEEE Transactions on Robotics* (pp. 1498-1511).

[30] Quaglia, G., Visconte C., Scimmi, L.S., Melchiorre, M., Cavallone, P., & Pastorelli, S. (2019). Robot arm and control architecture integration on a UGV for precision agriculture. In: T. Uhl, editor. *Advances in Mechanism and Machine Science - Mechanism and Machine Science* (pp. 2339-2348). Switzerland: Springer Nature Switzerland AG.

[31] Quaglia, G., Cavallone, P., & Visconte C. (2019). Agri_q: Agriculture UGV for monitoring and drone landing. In: Gasparetto, A., Ceccarelli M., editors. *Mechanism Design for Robotics Mech-anism and Machine Science* (pp. 413-423). Switzerland: Springer Nature Switzerland AG.

[32] Cavallone, P., Botta, A., Carbonari, L., Visconte, C., & Quaglia, G. (2020, September). The Agri. q Mobile Robot: Preliminary Experimental Tests. In *The International Conference of IFToMM ITALY* (pp. 524-532). Springer, Cham.

[33] Cavallone, P., Visconte, C., Carbonari, L., Botta, A., & Quaglia, G. (2020, September). Design of the Mobile Robot Agri. q. In *Symposium on Robot Design, Dynamics and Control* (pp. 288-296). Springer, Cham.

[34] Quaglia, G., Visconte, C., Carbonari, L., Botta, A., & Cavallone, P. (2020). Agri. q: A Sustainable Rover for Precision Agriculture. In *Solar Energy Conversion in Communities* (pp. 81-91). Springer, Cham.

[35] Carbonari, L., Botta, A., Cavallone, P., Tagliavini, L., & Quaglia, G. (2021). Data-Driven Analysis of Locomotion for a Class of Articulated Mobile Robots. *Journal of Mechanisms and Robotics*, *13*(5), 050905.

[36] Botta, A., Cavallone, P., Tagliavini, L., Carbonari, L., Visconte, C., & Quaglia, G. (2021). An estimator for the kinematic behaviour of a mobile robot 2 subject to large lateral slip. In *Appl. Sci.* (pp. 1-5).

[37] Fussum, T. V., Lewis, G. N. (1981, January). A Mathematical Model for Trailer–Truck Jackknifing. *Article in SIAM Review* (Vol. 23, no. 1), DOI: 10.1137/1023006.

[38] Latombe, J. C., (1991). Robot motion planning, Boston. Kluwer.

[39] Lumelsky, V. J. (1989). Motion planning. In W. Spong, M. Vidyasagar, *Robot dynamics and control* (pp. 7-8). Wiley, New York.

[40] Dubins, L. E. (1957). On curves of minimal length with a constraint on average curvature, and with prescribed initial and terminal positions and tangents. *American Journal of mathematics* (pp. 497-516).

[41] Bertaggia, M. (2018). Confronto di algoritmi di pianificazione di traiettoria sampling-based per robot mobili. Master's thesis, Scuola di Ingegneria Industriale e dell'Informazione, Politecnico di Milano, Italy.

[42] Prof. Siciliano, B., (Anno accademico 2011/2012). Pianificazione del Moto. Appunti del Corso di robotica avanzata. Corso di Laurea Magistrale in Ingegneria dell'Automazione.

[43] Wallace, R. S., Stentz, A., Thorpe, C. E., Moravec, H. P., Whittaker, W., & Kanade, T. (1985, August). First Results in Robot Road-Following. In *IJCAI* (pp. 1089-1095).

[44] Coulter, R. C. (1992, January). Implementation of the Pure Pursuit Path Tracking Algorithm, CMU-RI-TR-92-01, The Robotics Institute Camegie Mellon University Pittsburgh, Pennsylvania.

[45] Campbell, S. F. (2007). Steering control of an autonomous ground vehicle with application to the DARPA urban challenge. PhD Thesis, Massachusetts Institute of Technology.

[46] Amidi, O. (1990, May). Integrated Mobile Robot Control. Master's Thesis, Dept of Electrical and Computer Engineering, CMU.